Modelling and control of large flexible spacecraft

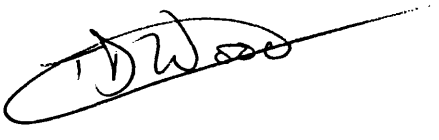
Wood, Timothy David

Award date:
1986

Awarding institution:
University of Bath

MODELLING AND CONTROL OF
LARGE FLEXIBLE SPACECRAFT

submitted by Timothy David Wood
for the degree of PhD
of the University of Bath 1986



ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

SUMMARY

The problems associated with modelling and control of large flexible spacecraft are examined. The representation of the dynamic behaviour of elastic structures in terms of normal modes is described, and forms the basis for two of the modelling techniques subsequently examined, that is hybrid co-ordinates, and finite elements. Also described is a technique based on a set of rigid bodies interconnected by elastic links.

Due to the infinite dimensional nature of the dynamics of flexible structures, the order of mathematical models is generally high, thus compounding the problems of control systems design. Consequently, various approaches for the design of controllers of practicable order are considered, and the effects on flexible structures of controllers designed in such a manner are also discussed.

The development of an experimental facility for demonstration of the practicality of such control schemes is described, including the experimental structure, actuators, sensors, and a dedicated multiprocessor digital computing system, with its various interfaces. The software to support the multiprocessor computer is described, together with mainframe software used to support the experimental facility.

The various techniques discussed for control systems design are applied to the problem of position control of the experimental structure. Simulation results are presented for the resulting closed loop systems to enable comparisions to be drawn. The suitability of the design methods for more complex cases is discussed, and subsequently such methods are applied to an example of a proposed large spacecraft which exhibits significant structural flexibility.

Two separate design exercises are considered for this case, the first having relatively low performance requirements, and the second having much higher performance requirements. Simulation results are again presented, which result in several conclusions and guidelines for control systems design for large flexible spacecraft.

Problem areas currently unresolved are identified, together with areas of interest for futher study.

ACKNOWLEDGEMENTS

The author would like to gratefully acknowledge the encouragement and advice of his academic supervisor Dr. B.A. White, and express his thanks to Prof. T. Rozzi, Chairman of the School of Electrical Engineering and Prof. J.F. Eastham, Head of the Control and Power Group, for the provision of School facilities. Thanks are also due to Prof. C.J. Harris, Chairman of the School of Electrical Engineering and Science, and Head of the Control and Guidance Group, of the Royal Military College of Science, Shrivenham, for the generous provision of School facilities during the latter stages of this work which was completed at R.M.C.S., Shrivenham. The assistance of staff and postgraduates at both establishments is most gratefully acknowledged.

The author would also like to express gratitude to his industrial supervisor Dr. M. Noton of British Aerospace, Space and Communications Division, Bristol, for his assistance and advice, and to his training officer Mr. R.H. Cripps of British Aerospace, Naval Weapons Division, Bristol, for his valuable support and for the provision of the NASTRAN finite element structural analysis facility. Thanks are also due to many other individuals at British Aerospace, Bristol, for their contributions to this work, in particular those who provided data and advice for modelling of the Space Platform.

The financial support of the Science and Engineering Research Council is also gratefully acknowledged.

Finally, the author would like to thank his wife Carol for her immense patience and support throughout the course of these studies.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

This thesis is concerned with the problems of modelling and control of large flexible spacecraft which are subject to three-axis stabilisation.

In the past, the flexibility of spacecraft which incorporate an active attitude stabilisation system has not been a particularly significant problem for the control systems designer, most designs being carried out as though the spacecraft was rigid. However, this is no longer generally possible, and as more adventerous projects are proposed, the demands on the control system performance are becoming ever more severe, and the control system design problem is becoming increasingly more difficult. For example, as the transmission requirements of communications satellites have increased, the size of the solar arrays used for generating power have also necessarily increased, resulting in lower natural vibration frequencies of the complete structure. At the same time, the accuracy requirements of the attitude control systems are becoming more stringent, often requiring increased control system bandwidth. As the bandwidth of controllers becomes closer to the resonant frequencies of the structure (or even overlap), the control system/structure interaction becomes problematical. It is shown that in these cases it is not generally possible to treat the spacecraft purely as a rigid body, and greater account has to be taken of the structural resonances.

Also, as spacecraft become larger, there is a greater requirement to keep the overall weight as low as possible in order to minimise the cost of launching the vehicle into space. Often the weight can be reduced at the expense of structural rigidity, but this obviously produces a structure with lower nat-

ural frequencies making system/structure interaction problems more likely.

This latter problem may be further compounded by different required objectives of the control systems. In the past, it has only been necessary to consider a single point of the structure because the substructure on which the actuators and sensors were mounted could be considered as being rigid, for instance spacecraft such as the NASA Skylab, and communications satellites such as the BAe Olympus. Although many future projects may also be treated in this manner, some of the spacecraft being proposed have structures with much more distributed flexibility, and may also have more complex control requirements such as shape control of large antenna reflectors. The problems of modelling and control of such spacecraft is the subject of this thesis.

1.2 Aims

There have been many papers published (mostly in the U.S.A.) in connection with this subject over the past few years, largely of a theoretical nature. An excellent literature survey, although a little out of date now, has been carried out by Burton and Rogers [1] which gives a good impression of the volume and range of literature associated with this subject. One of the primary aims of this work is to examine as much as possible of the published work in terms of its applicability to realistic problems and to present the information in a unified manner to enable a better understanding of the ideas presented and their inter-relationships.

One way in which the practicality of various ideas for control of a flexible structure can be investigated is to construct an experimental facility where control laws can be designed for a flexible structure constructed in a laboratory and subsequently implemented as a real-time control exercise. This approach has been adopted by some research centres in the U.S.A. (see for example [2], [3], [4]), and is also adopted here. However, due to the restrictions of an Earth-based enviroment, such laboratory structures are limited in their representation of complex spacecraft, and thus further studies are presented using a more representative example.

1.3 Chapter Contents

The subject of modelling of flexible structures is introduced in Chapter 2, starting with the description of elastic motion in terms of normal modes,

where both constrained and unconstrained modes are discussed. The hybrid co-ordinate approach is described, as it is currently the most popular technique for modelling the present generation of flexible spacecraft which can generally be thought of as a rigid body (where all the actuators and sensors are mounted) with large flexible appendages, which are primarily large solar arrays.

However, future spacecraft are less likely to comply with this general topology, but will exhibit a more flexible nature with the possibility of having significant flexibility in the part of the structure on which the control devices are mounted. In addition, there may be a requirement to actively control more than one point of the structure.

The need to be able to describe the motion of more than one point of the structure is catered for by the two modelling techniques subsequently discussed, the multibody approach, and the finite element approach. In both of these approaches, the infinite number of degrees of freedom of a flexible structure are represented by a finite number of elements each having a finite number of degrees of freedom. In the multibody case, each element is a rigid body with three degrees of freedom (rotation about each axis), and the structural flexibility is described in terms of elastic links between the rigid bodies. In the usual finite element case, elements are connected directly to each other and each element is described in terms of distributed mass and distributed stiffness properties.

All the modelling techniques produce mathematical models which have very high orders due to the large number of degrees of freedom of the physical representation. The inherently high order of mathematical models describing the dynamics of flexible structures compounds the problem of control system design.

In Chapters 3, 4, and 5 the problems of designing control systems for flexible structures are considered from an analytical viewpoint. Note that these three chapters are mutually dependent, and several terms are freely used which may be unfamiliar or appear ambiguous, so are now explained.

The term *controller* is used as a general reference to the entity, of whatever form, that is designed to make the plant (the spacecraft) behave as required. Also, two particular forms of controller are frequently referred to, these being a *state feedback based controller*, and an *output feedback based controller*. The fundamental difference implied by these terms is that the state feedback based controllers are designed assuming that the states of the system are available for feedback, even though they have to be reconstructed from the outputs of the system in practice, whereas the output

feedback based controllers are designed by considering the available outputs directly.

The term *state estimator* is used to refer to the entity which is used to reconstruct state information from the system outputs. Note that the term *state observer* is usually associated with deterministic systems, and the term *state estimator* is usually associated with stochastic systems. As will become apparent, the true "observation" of states, in the manner that the term *state observer* implies, is not possible for flexible structures, so the term *state estimator* has been adopted because it is felt that this term better implies the nature of the state reconstruction process for this class of problem.

The term *dynamic state estimator* is used to refer to state estimators which incorporate a dynamic system, and the term *static state estimator* is used to refer to state estimators which do not incorporate a dynamic system.

Chapter 3 considers various means of obtaining controllers with orders lower than the order of the mathematical models used to describe the system, such that they are a practical proposition for implementation. Chapter 4 considers the effects of feedback on flexible structures using both state feedback based controllers and output feedback based controllers, identifying terms associated with "spillover" as commonly used in the literature. This is an essential aspect of the application of controllers to flexible structures as any finite dimension mathematical model must inherently be an approximation to the real structure.

In Chapter 5 several methods of designing controllers are discussed, including optimal control, independent modal space control, modern modal control, and robust multivariable servomechanism control. Also discussed is the design of state estimators. Frequency domain techniques are referred to but not examined in detail for the reasons given, but another method which is examined is that of positive operators, referred to as Positivity, due to its application to spacecraft control problems as reported in the literature.

Chapters 6 and 7 are complementary and deal with the development of an experimental rig for the demonstration of proposed controllers to real-time application level. Various aspects of the hardware are discussed in Chapter 6, including the experimental structure itself, actuators, sensors, the multiprocessor digital computing system, and the various interfaces.

The software used to support this hardware is discussed in Chapter 7, including the software for the experimental rig computing system, and the software for the mainframe computer which provides additional support for the experimental facility.

Chapter 8 considers the application of the various methods of control system design introduced in earlier chapters to a relatively simple flexible structure, the experimental beam. The derivation of various mathematical models is detailed, together with the subsequent controller designs. Simulation results are presented to enable easy comparison between the resulting closed loop systems. As explained in Chapter 6, the experimental rig has not yet been developed sufficiently to enable closed loop control to be carried out and consequently it has not been possible to present results of real-time control exercises for comparison.

The results of Chapter 8 prompted refinement of the choice of control system design methods, and because the experimental beam structure was known to be only partially representative of a large flexible spacecraft, a second example was considered which is described in the following chapter.

Chapter 9 considers an early version of the ESA Space Platform as a realistic example of a large flexible spacecraft. The derivation of a mathematical model is described, followed by two controller design exercises, the second of which includes an examination of controllers based on a combination of state feedback and output feedback which are referred to as *dual basis controllers.* The results of these exercises are discussed and various observations are explained.

Chapter 10 presents some conclusions and guidelines resulting from these studies, and Chapter 11 suggests areas of interest for futher work.

Chapter 2

Mathematical Models of Flexible Structures

2.1 Introduction

It should be noted that the discussion in this chapter refers only to the problems of modelling structural flexibility of non-spinning three-axis stabilized craft. No account has been made of other problems such as mass movement due to fuel slosh, or personnel movement on a manned craft, although all of the techniques discussed may be extended to include such cases. Also, no account has been made of gravitational or orbital effects. For an initial study of the problems not examined here, see [5], [6], [7], [8], [9], or [10].

The next section of this chapter discusses the modal description of elastic motion, and subsequent sections examine various methods of using this modal description as part of the description of the overall motion of a flexible space structure. The overall motion of a flexible space structure can be described in a number of ways, depending on what assumptions are made about the structure, what details need to be described by the equations, and its topological layout. The first approach discussed is the "Hybrid Co-ordinate" approach which is the most popular technique for modelling current generation flexible space structures. However, due to its limitations, two other approaches are introduced, the "Multibody" approach, and the "Finite Element" approach. The final section in this chapter reviews the techniques discussed and suggests which particular approach is most suitable for a given problem.

2.2 Modal Description of Elastic Motion

A convenient way to describe the elastic (vibrational) motion of a structure is in terms of its modes [11],[12]. In the same way that a Fourier analysis of a random signal produces an infinite set of component sinusoidal signals [13], a modal analysis describes the elastic motion of a structure in terms of an infinite set of component signals called modes. One of the most convenient set of modes for vibration analysis is the set of "normal modes". Each normal mode is sinusoidal in nature, hence has a natural frequency, and a damping factor. The latter is usually very small and difficult to define, and thus is often neglected, or if included, is defined in a rather heuristic manner. Associated with each mode is a "modeshape" which describes the deformation of the structure for that particular mode. A time-dependent variable called a "modal co-ordinate" defines the magnitude of each mode with respect to time. Thus the product of a mode's co-ordinate and modeshape describe the displacement of the structure due to that mode. Therefore, the infinite sum of the products of modal co-ordinates and modeshapes describes the complete elastic motion of the structure. In practice, the analysis is usually discretized (for example, by using a finite element approximation), hence the number of degrees of freedom becomes finite, leading to a finite number of modes. This description stated mathematically in matrix notation is;-

$$d(t) = \Phi q(t)$$

where

$d(t)$ is the vector of structural co-ordinates (degrees of freedom)
$q(t)$ is the vector of modal co-ordinates
and Φ is the set of modeshape vectors i.e. $[\ \phi_1 \phi_2 \ldots \phi_n\ ]$
(where n is the number of modes)

The description of the elastic motion in terms of normal modes can provide some insight into the nature of the elastic motion in that it may be possible to identify certain modes which make negligible contribution to the overall motion. If such modes can be identified, it may be possible to remove these modes from the structural model. This is discussed in more detail in Chapter 3.

2.3 Hybrid Co-ordinate Approach

This approach was promoted by Likens and Bouvier [14] and Hughes [15],[16], and is based on a topological structure that consists of a rigid body to which flexible appendages are connected. (See figure 2.1.) This particular topology is convenient for most current generation spacecraft, where the main structure which carries all the sensors and actuators can be considered as completely rigid and the flexibibity is in the components connected to that structure, such as large solar arrays. Examples of space structures that have been considered to be of this form are the NASA Skylab craft [17], and communications satellites such as the B.Ae. Olympus [18].

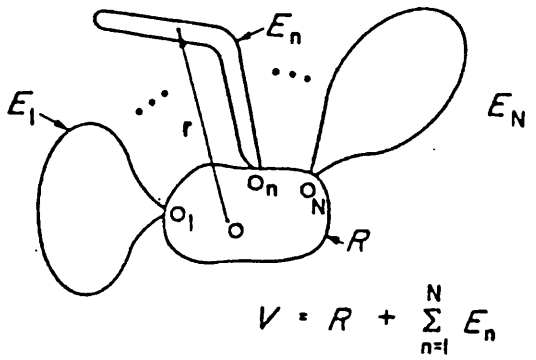


Figure 2.1: Schematic of Hybrid Co-ordinate representation

The name "hybrid co-ordinate" stems from the description of the rigid body attitude in terms of Euler angles and modal co-ordinates. It should be noted that there are two ways in which the equations of motion of such a structure can be derived, depending on the conditions imposed on the structure when obtaining the modal data. The first method is to analyse the flexible appendages in isolation, by considering the point at which each appendage is connected to the rigid body to be constrained so that no motion at all is possible. The subsequent modal analysis produces "constrained" modal data. These modes are sometimes referred to in the literature as "fixed-free", "appendage", or "cantilever" modes. When each appendage has been analysised, it is then necessary to evaluate the degree of coupling

between each constrained mode and the rigid body, these terms being known as the "coupling co-efficients". (Also referred to as "influence co-efficients".) A full derivation of the resulting equations of motion can be found in [15] or [16]. However, the equations are merely stated here in the form;-

$$\begin{aligned} I\ddot{\theta}(t) - \sum_{i=1}^{n} R^T \delta_i \ddot{\eta}_i(t) &= T(t) \qquad (a) \\ \ddot{\eta}_i(t) + 2\zeta_i \Omega_i \dot{\eta}_i(t) + \Omega_i^2 \eta_i(t) &= \delta_i^T R \ddot{\theta}(t) \quad i = 1, \ldots n \quad (b) \end{aligned} \tag{2.1}$$

where

I is the total spacecraft inertia,
$T(t)$ is the external torque on the rigid body,
$\theta(t)$ is the vector of rigid body attitude angles,
$\eta_i(t)$ is the *ith* constrained modal co-ordinate,
Ω_i is the frequency of the *ith* mode,
ζ_i is the damping factor of the *ith* mode,
δ_i is the coupling coefficient of the *ith* mode to the rigid body,
R is the transformation matrix between the appendage axis system and the rigid body axis system,
and n is the number of modes in the model.

The second method applies no constraints to the structure at all. The modal analysis is applied to the complete structure to obtain "unconstrained" modal data, sometimes referred to in the literature as "free-free", "global", "system", or "vehicle" modes. The modal analysis in this case produces the coupling coefficients automatically, as they are a function of the value of a mode's shape at the rigid body co-ordinate. Again, a full derivation of the resulting equations of motion can be found in [15] or [16]. They are merely stated here in the form;-

$$\begin{aligned} I\ddot{\theta}_r(t) &= T(t) \qquad (a) \\ \ddot{\theta}_{fi}(t) + 2\zeta_{fi}\omega_{fi}\dot{\theta}_{fi}(t) + \omega_{fi}^2 \theta_{fi}(t) &= T(t) K_{fi}/I \qquad (b) \\ \theta(t) &= \theta_r(t) + \sum_{i=1}^{n} \theta_{fi}(t) \quad i = 1, \ldots n \quad (c) \end{aligned} \tag{2.2}$$

where

I is the total spacecraft inertia,
$T(t)$ is the external torque on the rigid body,
$\theta(t)$ is the vector of rigid body attitude angles,
$\theta_r(t)$ is the vector of contributions due to the rigid modes,
$\theta_{fi}(t)$ is the contribution due to the *ith* elastic mode,
($\theta_r(t)$ & $\theta_{fi}(t)$ form the unconstrained modal co-ordinates)
ω_{fi} is the frequency of the *ith* elastic mode,
ζ_{fi} is the damping factor of the *ith* elastic mode,
K_{fi} is the coupling co-efficient of the *ith* elastic mode to the rigid body,
and n is the number of modes in the model.

Note that as the appendages and the rigid body are analysised simultaneously, it is necessary to use just one axis set for the complete structure, thus avoiding the need for axis transformation matrices as used in the constrained method as described earlier.

Although the constrained method appears to produce a simpler set of equations, care should be exercised over its use, as the equations are in terms of modal frequencies of the appendages alone, whose frequencies are not generally the same as the modes of the complete structure as obtained by the "unconstrained" method. This is of particular importance if model order reduction is anticipated, as the choice of analysis method can significantly affect the accuracy of the resultant reduced order model. The reason for this is now discussed.

Although an infinite number of modes are required for complete description of the elastic motion of a structure, in practice usually only a subset of these are used to describe the motions for reasons that will become apparent later. The actual choice of this subset is discussed in detail in Chapter 3. However, if the selection procedure is applied to the "constrained" set of equations, when the control system design model is evaluated (which will contain information on system frequencies, not appendage frequencies), the system modal information is derived from a set of constrained modal data which may be significantly depleted. It is often assumed that the first system mode only depends on the first appendage mode, but Hablani [19] and Likens *et al.* [20] have shown that this is not in general a valid assumption, and that it can lead to very large inaccuracies in derived system frequencies (see also [21]).

This problem does not occur when reducing the order of the unconstrained equations as the unconstrained approach produces the system frequencies directly. Hence, if it is desired to reduce the order of hybrid co-ordinate equations by mode deletion, it is advisable to apply this procedure to system modes only, either by direct application to the unconstrained equations, or by converting the constrained equations to unconstrained equations using a full set of constrained modes to avoid the introduction of unnecessary inaccuracies in the system model.

These inaccuracies stem from the conditions under which the constrained modal data is obtained, that is the rigid body is constrained to be motionless. If the spacecraft under analysis has a rigid body with an inertia which is very much greater than the inertia of its appendages, then this condition will almost be satisfied, and thus the system frequencies will be not be very much different from the appendage frequencies. In this case deletion of constrained modes may not introduce significant inaccuracies in the derivation of a system model.

It can be shown mathematically (see Appendix A.2), that the addition of constraints effectively increases the stiffness of the system, which results in higher modal frequencies. Hence constrained modal frequencies will always be higher than system modal frequencies.

2.4 Multibody Approach

This approach was promoted primarily by Hooker and Margulies [22] and later further developed by Hooker [23]. It is based on a structure described by a set of n rigid bodies interconnected by dissipative elastic joints and subject to arbitrary torques. The only requirement on the configuration is that it should be equivalent to a topological tree, that is it should have no closed loops. (See figure 2.2). The applicability of this topology is, perhaps, not quite so obvious as that of the hybrid co-ordinate approach described above. However, it has been used for modelling of boom antennae and fold-out solar arrays during deployment [24], [25].

The full derivation of the equations of motion for a structure described in such a manner can be found in [24], they are merely stated here in the form;-

$$A\ddot{\theta}(t) - PC(t) = T(t) \tag{2.3}$$

where

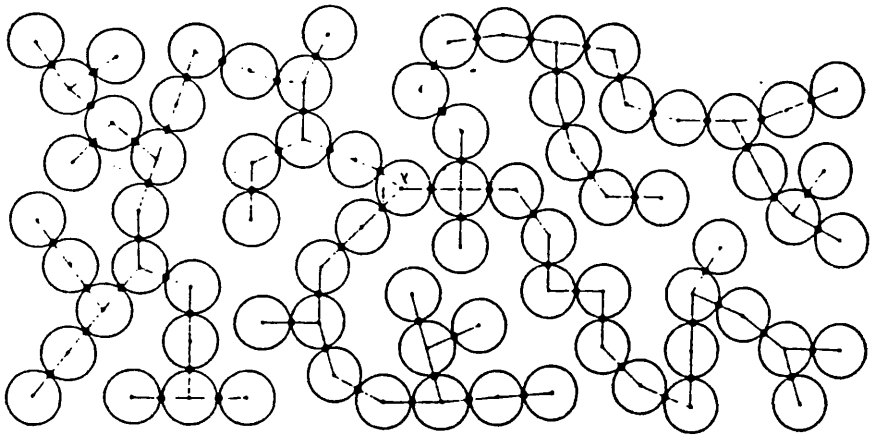


Figure 2.2: Schematic of multibody representation

A is the matrix of inertia properties of the bodies,
$\theta(t)$ is the vector of attitude angles of the bodies,
$T(t)$ is the matrix of external torques on the bodies,
P is the matrix of vectors describing the "locked modes" of the structure (see [22]),
and $C(t)$ is the vector of constraint torques between bodies.

The "locked modes" referred to above are modes where certain degrees of freedom are constrained. It should be noted that only rotational motion is permitted at the elastic joints so it is not convenient to describe any translational motion within the structure. However, an advantage of this method is that it can cope equally well with large deformations as it can with small deformations, hence its use for modelling the deployment of fold-out solar arrays in [24] and [25].

2.5 Finite Element Approach

This approach, although having classic origins [11],[12], albeit in a slightly different form, has only been applied to flexible space structures in recent years. It is based on the description of a structure in terms of a finite number of co-ordinates (degrees of freedom) connected by elements with distributed mass and stiffness properties. This technique has been used for many years

in the aircraft industry for the analysis of static and dynamic loading of airframes [26],[27],[28], and other problems such as structural "flutter" [29]. A particularly convenient facet of the finite element method is that there are few (only computational) limitations as to how many elements need to be used, or what configurations are permissible, so a crude representation with a small number of elements may be utilised for initial studies, with refinement of the representation as more accurate data becomes available as the spacecraft design progresses.

The classical solution of this type of problem is via methods such as the Rayleigh-Ritz method, or the Galerkin method. A modern alternative is the finite element method, which can be considered as a combination of these two classic methods as it uses assumed shape functions to represent displacements within element boundaries [30],[31].

The classic derivation of the equations of motion for a structure modelled in this manner can be found in most texts on structural vibration analysis (for example, see [11],[12],[30],[31]), but the derivation of these equations using assumed shape functions is perhaps less well known and thus full details are given in Appendix A.1, together with details of decoupling the equations into modal form. However, they are stated here in the form;-

$$M\ddot{d}(t) + D\dot{d}(t) + Kd(t) = F(t) \tag{2.4}$$

where

$d(t)$ is the vector of structural co-ordinates (degrees of freedom),
$F(t)$ is the vector of external forces and torques at the structural co-ordinates,
M is the matrix of mass and inertia properties of the elements,
D is the matrix of damping properties of the elements,
K is the matrix of stiffness properties of the elements.

Note that the damping terms are generally very small and ill-defined, so are neglected for many cases.

These coupled equations of motion can be decoupled by a suitable co-ordinate transformation. The most convenient transformation is to the second order modal form, which is described in detail in Appendix A.1. The resulting modal matrix, appropriately normalised, can be used to map the structural co-ordinates into modal co-ordinates thus enabling equation 2.4 to be written as a set of uncoupled equations of the form;-

$$\ddot{q}_i(t) + 2\zeta_i\omega_i\dot{q}_i(t) + \omega_i^2 q_i(t) \quad = \quad f_i(t) \quad i = 1, \ldots n \tag{2.5}$$

where

$q(t)$ is the *ith* modal co-ordinate,
ω_i is the frequency of the *ith* mode,
ζ_i is the damping factor of the *ith* mode,
$f_i(t)$ is the generalized force at the *ith* modal co-ordinate,
and n is the number of modes. (equal to the number of degrees of freedom of the model)

An advantage of this approach is that there is no requirement for any point of the structure to be rigid. The use of a finite element structural analysis package for a digital computer such as NASTRAN [32], [33] enables very complex structures to be modelled, including non-linear elements such as sliding joints. Most finite element structural analysis programs include a dynamic analysis capability which provides the modal frequencies and the correctly normalized modeshape matrix (see Appendix A.1) such that the uncoupled modal equations can be derived directly without further computation.

Note that translational and rotational motion can be described for the elastic modes as well as the rigid modes (where the complete structure moves with no deformation from its nominal shape) hence the complete motion of the structure can be described in terms of modes.

In addition, one particular advantage of this type of model is the facility to model displacements at more than one point with ease, thus allowing the examination of effects of structural flexibility between control devices (actuators and sensors) rather than having to assume that there is no relative displacement between control devices. This enables the examination of "shape" control problems where it is required to minimise the relative displacements between a set of points on a structure. Examples of this type of problem might be a large antenna, where flexing needs to be minimised in order to minimise the deformation of the radiation pattern of the antenna, or a large optical telescope where deformations of the line-of-sight have to be minimised in order to maximise the resolution of the telescope.

The only apparent limitation to the finite element approach is that it is based on an assumption of small displacements. Thus if the motions are small, the rigid modes can be handled together with the vibration modes. When larger rigid mode motions are to be considered, and motion rates

remain small, the equations are still applicable provided that the rigid modes are now given in terms of attitude angles, and the elastic displacements are now interpreted as deformations of the structure relative to the rotated frame defined by the rigid modes. (The equations take on a similar form to those given for the hybrid co-ordinate approach via the unconstrained method (equations 2.2.) Note that the elastic deformations must always remain small though, because for large elastic deformations other non-linear effects also have to be modelled.

2.6 Summary

The choice of modelling technique for a particular problem is likely to be influenced by the general topology of the structure, by the control requirements, and in some cases, by the magnitude of the motions of the structure that are required to be modelled.

For structures which can be essentially considered as a rigid body with flexible appendages, where all the sensors and actuators are mounted on the rigid body, and where the only control requirement is the "pointing" (that is position control) of that rigid body, then the hybrid co-ordinate approach is likely to be the most suitable method. An example of such a structure might be a large geostationary communications satellite.

For reasons already stated in the section on the hybrid co-ordinate approach, it is recommended that the modal analysis is carried out on the complete structure with no artificial constraints, rather than individually analysising the appendages by using false constraints, that is use the unconstrained method in preference to the constrained method.

For cases where the structural flexibility is more distributed, or if the control requirements apply to more than one point of the structure, then another approach is necessary. If only rotational motion is important, then the multibody approach may be a suitable method. However, in many cases the problems of obtaining data for the elastic joints might prove difficult, and in such cases the finite element approach may be preferable.

The finite element approach may, to some extent, be considered as being more generalized than the hybrid co-ordinate and the multibody approaches. This is due to the fact that the finite element approach can be used to represent both translational and rotational vibratory motion at more than one point on a structure which exhibits distributed flexibility. There is no need to be able to define any part of the structure to be rigid, although this

is easily accommodated if required.

Another advantage of the finite element approach is the ease with which the structural stresses and strains can be computed. (See Appendix A.1.) Indeed, in the aircraft industry it has been common practise for many years to use a finite element model to investigate the static and dynamic loading of an airframe, as well as obtaining modal data for vibration analysis. This may be useful for large complex structures where the stresses on the structure may be computed which would then allow lightly stressed components to be identified and redesigned for a lower factor of safety, generally resulting in a weight reduction, an aspect of significant importance in the design of large spacecraft.

The main limitation of the finite element approach in its basic form, is that it is based on an assumption of small displacements. For large angles and large rates, other non-linear effects have to be modelled, hence for these cases, the multibody approach is probably more useful, as it does not have any small displacement limitations. An example of this is where the structure undergoes significant changes, such as the unfolding of fold-up solar arrays, or perhaps the on-orbit assembly of large spacecraft.

Thus when choosing a modelling technique for application to a space structure, it may be necessary to use different techniques to represent the same structure according to the particular problem being dealt with.

Another possibility which has not been examined here is to combine the multibody approach and the finite element approach such that the individual bodies of the multibody description can exhibit elastic behaviour. The small deformations of the bodies could be described by finite element equations in local body co-ordinates, whilst the interactions of the bodies could be described using the multibody equations. This would facilitate the simultaneous examination of the effects of large structural changes and elastic deformations.

The mathematical models derived can be written in the usual transfer function matrix formulation, or in state space formulation, as preferred, for control system design. A disadvantage of the transfer function description is the large peaks in the transfer functions at the structural frequencies caused by the low structural damping. This can cause problems in observing detail in transfer function plots generated by computer programs using automatic scaling. The basis for a state space description can be chosen to correspond conveniently with the second order modes of the system (see Appendix A.1), and its matrix structure is most convenient for manipulation on a digital computer.

Because of its convenient form, particularly in a state space representation, the modal form of the finite element representation (equation 2.5) is used in subsequent chapters, and so this form of the equations are stated here as follows;-

$$\dot{x}(t) = Ax(t) + Bu(t)$$

$$y(t) = Cx(t)$$

where $x \in \mathcal{R}^n$ is the vector of modal amplitudes and rates, $u \in \mathcal{R}^m$ is the vector of external forces and torques, and $y \in \mathcal{R}^p$ is the vector of nodal displacements and rates. Matrix A is block diagonal, with blocks of the form;-

$$a_i = \begin{pmatrix} 0 & 1 \\ -\omega_i^2 & -2\zeta_i\omega_i \end{pmatrix}$$

and matrix B has corresponding blocks of the form;-

$$b_i = \begin{pmatrix} 0 \\ \Phi_i^T \end{pmatrix}$$

and matrix C also has corresponding blocks, which have the form;-

$$c_i = \begin{pmatrix} \Phi_i & 0 \\ 0 & \Phi_i \end{pmatrix}$$

Note that Φ_i is the "shape" of mode i at the actuator and sensor locations.

Chapter 3

System Order Reduction

3.1 Introduction

The design and implementation of controllers for high order systems is plagued with difficulties. Firstly, the design of controllers for high order systems tends to be complicated by numerical problems, such as poor convergence, or even non-convergence, of iterative algorithms. Also, the computation costs of design algorithms generally increases markedly with increasing system order. Secondly, even if a design algorithm successfully produces a controller specification, the resulting controller can be of similar order as that of the system, which may be too complex for real-time implementation with the available processing power. (This is particularly likely when using state feedback techniques, because the states are not measurable and so a dynamic estimator will, in general, be required.) The general problems of designing controllers for high order systems has been the subject of much work reported in the literature, and some aspects of these problems are now discussed.

There are basically three approaches to the problem of designing low order controllers for high order systems, these being;-

1. Reduce the order of the system model, then design a controller for the low order model.

2. Design a controller for the high order system model, reduce the order of the controller to meet implementation restrictions.

3. Design a controller for the high order system model with constraints such that a controller is produced which is of low enough order to meet

implementation restrictions.

The first approach has received considerable attention in the literature (some particularly noteworthy examples being [34],[35],[36],[37], and [20], no doubt because of the computational attractiveness of applying design algorithms to low order models rather than high order models. The second method has been much less well examined, probably for two reasons. The first reason is that the design of high order controllers is unattractive because of computational costs and numerical difficulties, and the second reason is that many of the methods postulated for model order reduction could also be applied to the problem of controller order reduction. The third method has been comparitively seldom reported, although as will become apparent, it may be the most technically desirable method.

3.2 Model Order Reduction

Many techniques have been postulated for model order reduction in both the frequency domain and the time domain, the inter-relation of several of these is shown in figure 3.1. It is shown in [17, chapters 2 and 3] that many of these techniques are merely special cases of other more general approaches. As all the control system design methods discussed later use the state space description [38], that is the time domain, then model reduction methods for the time domain only are considered further.

As can be seen from figure 3.1, the time domain methods can be grouped under two headings, perturbation and aggregation. The perturbation methods can also be sub-divided into two groups, non-singular and singular (sometimes termed weakly-coupled and strongly-coupled). Non-singular perturbation model reduction can be used where parts of a system are only weakly coupled, a situation which does not generally occur in large flexible spacecraft. (Except perhaps for the separation of the three principal axes of a non-spinning spacecraft where the cross-products of inertia are small.) Singular perturbation model reduction can be used where a part of the system has much faster dynamics than that of the remainder, so that the remainder is dominant. This technique can be considered as a valid technique only when the slowest part of the high frequency portion is about three to five times (or more) faster than the fastest part of the low frequency section. Again, this situation does not generally occur in the dynamics of large flexible space structures. However, a method has been reported [39]

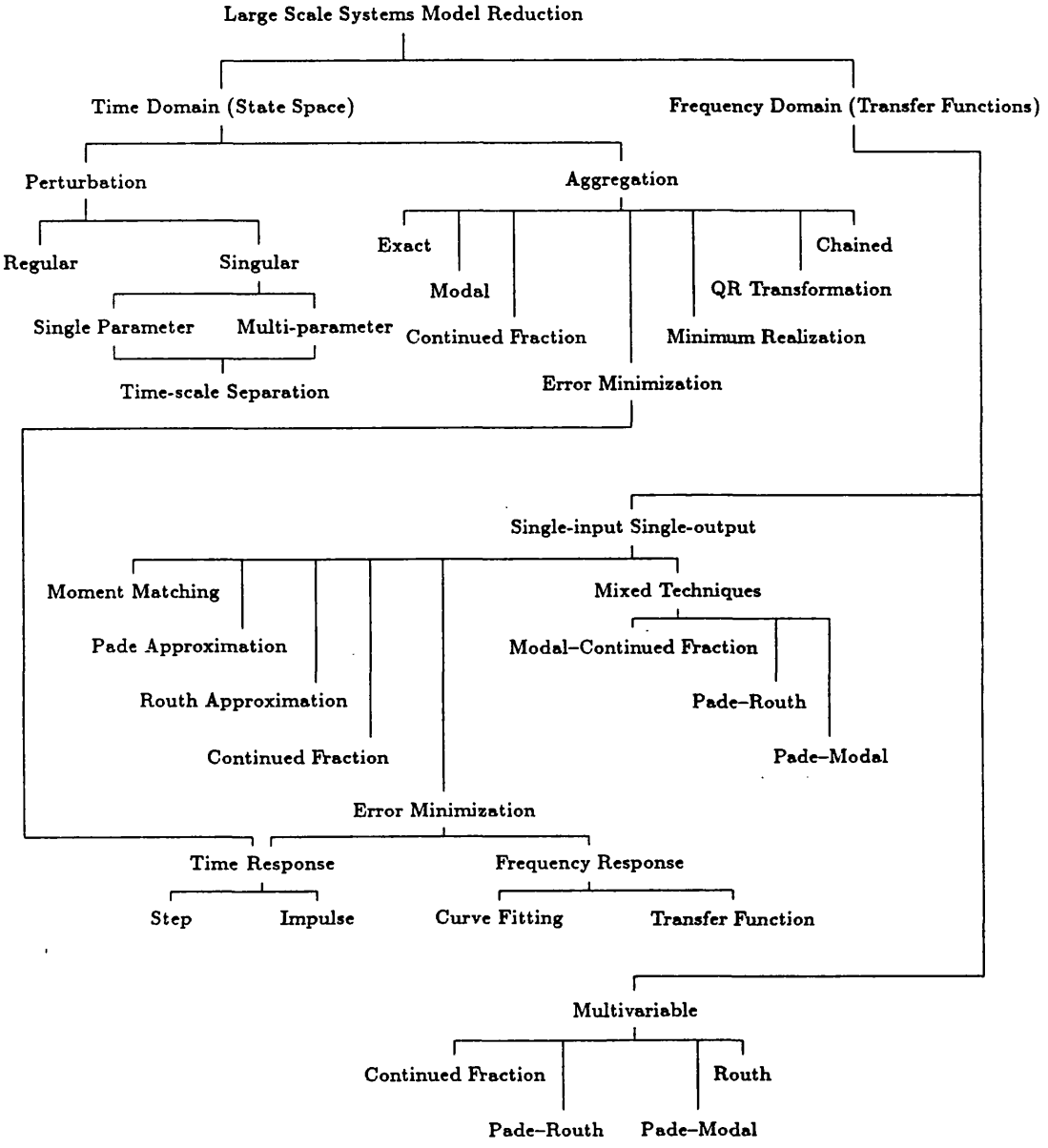


Figure 3.1: Inter–relation of system reduction methods

where "forced singular perturbation" has taken place by using a suitable control law, and this is examined in Chapter 5.

Model reduction via aggregation can take various forms but it is essentially the choice of a new set of system states (either completely new states or a subset of the original states) which approximates the original system behaviour. This can be considered mathematically as a co-ordinate transformation from a *nth* order space to a *mth* order subspace, where $m < n$. The difference between the various methods is the means of obtaining the transformation matrix. It has already been noted that many of the reported methods are merely special cases of a more generalized approach, but where a particular method is used to obtain the transformation matrix, or where a particular form for the transformation matrix is assumed. (Although many of the methods have not been reported by their authors in such a manner.)

For example, one method of forming the transformation matrix is via the minimization of an error function, which in some manner describes the error between the original model and the reduced order model [40]. This approach has certain drawbacks, such as the definition of a suitable error function for minimizing, and the possible computation problems of minimizing a high order function, subject to a large number of constraints.

Another example is the method of model reduction by Component Cost Analysis [34],[35]. This technique assigns a "cost" to components of a system, thus identifying the components with highest "cost" as those which should be retained in a low order model. The remaining components are simply ignored. Thus an *nth* order system described by a matrix equation of the form;-

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned}$$

can be approximated by a *mth* order model (where $m < n$) of the form;-

$$\begin{aligned} \dot{x'}(t) &= A'x'(t) + B'u(t) \\ y(t) &= C'x'(t) \end{aligned}$$

where;-

$$x'(t) = Lx(t) = \begin{pmatrix} I_m & 0 \end{pmatrix} \begin{pmatrix} x_r \\ x_d \end{pmatrix}$$

I_m is an *mth* order identity matrix, x_r represents the retained states (corresponding to the retained components) of the original *nth* order system,

and x_d represents the deleted states (corresponding to the deleted components of the original system). Thus a particular form for the transformation matrix is assumed, only the selection of components, that is the ordering of the system states into retained or deleted groups, is selected by the reduction method.

When Component Cost Analysis is used for model reduction, a suitable co-ordinate basis has first to be selected, and then a definition for the "costs" has be to chosen.

For flexible structure models, a particularly convenient co-ordinate basis for the definition of the system components is the modal basis, where the components analysised are the second order normal modes of the system corresponding to the vibration frequencies of the structure, and this is generally referred to as Modal Cost Analysis [35],[36],[41].

The "cost" of a component is generally defined as its contribution to a quadratic cost function, for example;-

$$V = \int_0^\infty (y^T y) dt$$

This is particularly convenient for flexible spacecraft, because when the damping is very light, as is the case for flexible spacecraft, this "cost" definition for each mode simplifies to the product of the mode's controllability norm, its observability norm, and its time constant [35],[36]. Thus the computation of the "cost" for each mode is simple and quick which makes this method most suitable for very high order systems.

In addition, some measure of the "quality" of the reduced order model can be computed. This is known as the "model error", and is defined by the expression;-

$$\text{model error} = \frac{V_d}{V_r + V_d}$$

where V_d is the cost of the deleted modes and V_r is the cost of the retained modes.

Note that the denominator is the sum of the costs of all the modes in the system, and the numerator is the cost of all the modes which are discarded when forming the reduced order model, so the "model error" can be interpreted as some measure of the amount of system information which has been discarded in the formation of the reduced order model.

Modal Cost Analysis is thus a very convenient method for reduction of high order models of flexible structures. The computation of the "cost" of each mode can be easily programmed on a digital computer, thus allowing

its application to systems of very high order without any numerical difficulties. By ordering the modes in terms of "cost", a reduced order model can be formed from the modes with highest "cost", sucessively increasing the number of retained modes until the "model error" is sufficiently small.

It should be noted, however, that the "cost" of a mode is dependent on its time constant, which in turn depends on the damping, as well as the frequency, of that mode. In the analysis of flexible spacecraft, damping terms are ill-defined, hence frequently, a rather arbitrary value of damping is assumed. If all the modes are assumed to have the same level of damping, then this should not be a problem, because although the mode "cost" will depend on the absolute value of the damping, the "model error" will be independent of the value of damping, so the selection of modes for a reduced order model will be the same for a given system, irrespective of the assumed value of damping, provided the assumed value is greater than zero. If a mode has no damping, the time constant of that mode, and hence its "cost", will be infinite. Hence, Modal Cost Analysis suggests that any rigid modes of a system should always be retained in a reduced order model, and that if all the elastic modes are assumed to have no damping, then none of the modes should be deleted, that is a reduced order model should not be formed.

3.3 Full Order Design – Controller Reduction

This approach for obtaining a low order controller has not been well supported in the literature, probably for the reasons mentioned in section 3.1 However, one notable method that has been reported for controller reduction is the algorithm presented by Yousuff and Skelton [42], referred to as the "Q-cover" algorithm, which attempts to match the q Markov parameters of a qth order controller with the first q Markov parameters of the full order controller, and in addition, also attempts to maintain the covariances of the system outputs.

Thus this approach attempts to derive a low order controller which approximates the control action of the full order controller. The algorithm is only outlined here as full details of the algorithm can be found in [42].

Consider a nth order system described by the following equations;-

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) + Dw(t) \\ y(t) &= Cx(t) \\ z(t) &= Mx(t) + v(t) \end{aligned}$$

where $x \in \mathcal{R}^n$ and $u \in \mathcal{R}^m$. The disturbances $w \in \mathcal{R}^d$ and $v \in \mathcal{R}^l$ are assumed to be uncorrelated (with each other and with $x(0)$) zero-mean white noise processes with intensities $W > 0$ and $V > 0$ respectively. The vector $z \in \mathcal{R}^l$ is composed of the measurements corrupted by the noise $v(t)$. The vector $y \in \mathcal{R}^k$ contains only the variables that are used to measure the performance of the system via the standard quadratic cost function $\mathcal{V}$ defined as;-

$$\mathcal{V} = \mathcal{E}\left[\lim_{t\to\infty} \frac{1}{t}\int_0^t \left\{\|y(\tau)\|_Q^2 + \|u(\tau)\|_R^2\right\} d\tau\right] \tag{3.1}$$

where $Q > 0$ and $R > 0$ are weighting matrices and $\mathcal{E}$ denotes the expectation operator.

Assuming that the matrix pairs (A, B) and (A, D) are controllable, and the matrix pairs (A, C) and (A, M) are observable, the LQG controller (denoted by $\mathcal{S}_c$) that minimises equation 3.1 is given by;-

$$\begin{aligned} \dot{x}_c(t) &= A_c x_c(t) + F z(t) \\ u(t) &= G x_c(t) \end{aligned}$$

where $x_c \in \mathcal{R}^n$ and;-

$$\begin{aligned} A_c &= A + BG - FM \\ F &= PM^T V^{-1} \\ G &= -R^{-1}B^T K \end{aligned}$$

with;-

$$\begin{aligned} KA + A^T K - G^T RG + C^T QC &= 0 \\ PA^T + AP - FVF^T + DWD^T &= 0 \end{aligned}$$

Given $\mathcal{S}_c$, it is required to obtain a controller of order $r < n$, denoted by $\mathcal{S}_r$, in the following form;-

$$\begin{aligned} \dot{x}_r(t) &= A_r x_r(t) + F_r z(t) \quad (a) \\ u(t) &= G_r x_r(t) \qquad\qquad\quad (b) \end{aligned} \tag{3.2}$$

where $x_r \in \mathcal{R}^r$.

The matrices A_r, F_r, and G_r are obtained via an oblique projection of the LQG-controller $\mathcal{S}_c$, such that;-

$$A_r = L_r A_c T_r$$

$$\begin{aligned} F_r &= L_r F \\ G_r &= G T_r \\ L_r T_r &= I_r \end{aligned}$$

where $L_r \in \mathcal{R}^{r \times n}$, $T_r \in \mathcal{R}^{n \times r}$, and I_r denotes the $r \times r$ identity matrix. The matrices L_r and T_r are obtained via the following algorithm.

1. Specify q, the number of Markov parameters to be matched by the reduced controller.

2. Compute X_c the steady-state covariance of the states x_c by solving;-

$$X_c(A + BG)^T + (A + BG)X_c + FVF^T = 0 \tag{3.3}$$

3. Construct the qth observability matrix Θ_q defined by;-

$$\Theta_q^T = \left[G^T, A_c^T G^T, \cdots, (A_c^{q-1})^T G^T\right] \tag{3.4}$$

4. Compute the singular value decomposition of $\Theta_q X_c \Theta_q^T$ as follows;-

$$\Theta_q X_c \Theta_q^T = [U_1 U_2] \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} U_1^T \\ U_2^T \end{bmatrix} \tag{3.5}$$

where;-

$$\begin{aligned} \Sigma &= diag(\sigma_1^2, \sigma_2^2, \cdots, \sigma_{r_q}^2) \\ r_q &= rank\left[\Theta_q X_c \Theta_q^T\right] \\ U_1 U_1^T + U_2 U_2^T &= I_{qm} \end{aligned}$$

and $U_1 \in \mathcal{R}^{m \times r_q}$.

5. Construct the projection matrices L_r and T_r from;-

$$\begin{aligned} L_r &= U_1^T \Theta_q \\ T_r &= X_c L_r^T \left(L_r X_c L_r^T\right)^{-1} \end{aligned}$$

Thus, when L_r and T_r have been computed, the reduced order controller can be constructed, as defined by equation 3.2. Note that the *ith* Markov parameter $\mathcal{M}_i$ of the reduced order controller is given by;-

$$\mathcal{M}_i = G_r A_r^{i-1} F_r$$

Some comments regarding the computational aspects of this algorithm are now considered.

1. The solution of equation 3.3 can be found using any one of a number of algorithms for solving a Lyapunov type equation, (see Appendix B.4) so assuming the choice of a reliable algorithm, this is unlikely to cause many problems.

2. The construction of the observability matrix Θ_q (equation 3.4) may cause problems due to the raising of matrix A_c up to the power of q. Even if A_c is only moderately ill-conditioned initially, the subsequent products will, in general, cause further deterioration of the condition of the final matrix. This will be further compounded by the computation of the product $\Theta_q X_c \Theta_q^T$ in equation 3.5.

3. Reliable algorithms for the computation of the singular value decomposition of a matrix are readily available, so this should not cause any problems, except perhaps with an ill-conditioned matrix, which could occur quite easily, as mentioned above.

It is beyond the scope of this work to examine any numerical difficulties in detail, so other than scaling or simple balancing techniques, no attempts were made to overcome any numerical problems when the above algorithm was implemented on a digital computer. Consequently, numerically difficult problems sometimes caused the algorithm to fail. More recently, Skelton and Collins [43] have reported a method of obtaining a class of reduced order controllers where portions of the projection matrices are parameterised, thus reducing the computational burden.

It should be noted that reduced order controllers obtained by this method of matching Markov parameters are not guaranteed to be stable, nor is the resulting closed loop system using a reduced order controller derived in this manner guaranteed to be stable.

As was mentioned in section 3.1, some of the methods for model reduction can also be applied to the problem of controller reduction. One such method that has been reported in the literature is Modal Cost Analysis [36]. The basis of this approach is the same as that described in section 3.2 for model reduction with the difference that the "cost" is defined in a manner more appropriate for a closed loop system.

In particular, if the controller is designed by solving an optimal control problem with a quadratic cost function of the form;-

$$V = \int_0^\infty (y^T Q y + u^T R u) dt$$

then an obvious choice for a cost function for Modal Cost Analysis of the controller would be the same function. Thus the modes of the controller can be analysised in terms of their contribution to the cost function and modes exercising least effect can be deleted from the controller.

The derivation of modal costs for controller reduction given in [36] effectively identifies those states whose feedback contributes least to the performance of the closed loop system (as defined by the cost function), thus allowing the columns of the state feedback gain matrix corresponding to these states to be deleted. It is then necessary to design a state estimator to produce estimates of the retained states.

Also reported in [36] are equations defining the pole shifts due to the controller reduction. These equations, however, are based on the assumption of perfect state measurements, and as is shown in the next chapter, the pole shifts caused by the employment of a state estimator may be considerably larger.

3.4 Low Order Controller directly from High Order Model

This last approach is the most desirable method from a design viewpoint, as there are no approximations necessary in the derivation of the controller. (The two approaches discussed in sections 3.2 and 3.3 incorporate an approximate model or an approximate controller, respectively). Of course, one method of obtaining a low order controller for a high order system is to use a controller which is synthesised directly from the output measurements, rather than use a state feedback law which requires a dynamic state estimator to reconstruct the unmeasurable or unobservable states. (In the modal description, none of the states of a flexible spacecraft model can be measured as they are not physical variables.) Such a method has been reported in [44], which is based on the so-called Robust Servomechanism Problem [45], where for a system of the form;-

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned}$$

a control law can be synthesised of the form;-

$$u(t) = -F_1 y(t) - F_2 \dot{y}(t) - F_3 \int (y_{ref} - y(t)) \qquad (3.6)$$

The elements of the gain matrices are obtained using a constrained parameter optimization method [46],[47],[48]. Although high order parameter optimization problems are not computationally attractive, this method has been applied to a large flexible spacecraft example which was 200th order [44]. However, the controller design for the example reported in [44] was not based on the full 200th order model, but the authors took advantage of the predictable stability properties of output feedback (discussed in detail in Chapter 4), such that the model used for controller design comprised the rigid modes only, that is the elastic modes were ignored completely.

It should be noted that conditions have been derived [45],[44] for the existance of stabilizing controllers, such that it is possible to decide whether a "three-term" controller (as equation 3.6) is sufficient to meet the design requirements, or whether it is necessary to use a more complex controller based on state feedback with a state estimator.

A method of obtaining a low order controller incorporating a dynamic state estimator and the use of state feedback has been reported by Hyland and Bernstein [49],[50], which is known as the "optimally projected equation" method. This method effectively requires the solution of the controller design equations for the high order system with the constraints of the low order controller design equations. This manifests itself as the simultaneous solution of two modified Riccati equations and two modified Lyapunov equations, a non-trivial problem. Full details of the algorithm can be found in [49] and [50] and a comparison of this method with other approaches can be found in [51]. The algorithm is outlined in the following.

For a system of the form;-

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) + H_1 w(t) \\ y(t) &= Cx(t) + H_2 w(t) \end{aligned}$$

it is required to obtain a fixed order controller of the form;-

$$\begin{aligned} \dot{x}_c(t) &= A_c x_c(t) + B_c y(t) \\ u(t) &= C_c x_c(t) \end{aligned}$$

which will minimise the performance criteria;-

$$J = \int_0^\infty (x^T R_1 x + u^T R_2 u) dt$$

Suppose (A_c, B_c, C_c) solves the steady-state fixed order controller problem, then there exists matrices $Q, P, \hat{Q}, \hat{P}, G, M$, and Γ such that;-

$$\hat{Q}\hat{P} = G^T M \Gamma$$

$$\Gamma G^T = I$$

and A_c, B_c, and C_c are given by ;-

$$\begin{aligned} A_c &= \Gamma\left(A - Q\overline{\Sigma} - \Sigma P\right) G^T \\ B_c &= \Gamma Q C^T (H_2 H_2^T)^{-1} \\ C_c &= -R_2^{-1} B^T P G^T \end{aligned}$$

for some (G, M, Γ) factorisation of $\hat{Q}\hat{P}$, and such that with $\Upsilon = G^T\Gamma$ the following conditions are satisfied;-

$$0 = AQ + QA^T + H_1 H_1^T - Q\overline{\Sigma}Q + \Upsilon Q\overline{\Sigma}Q\Upsilon^T \quad (3.7)$$

$$0 = A^T P + PA + R_1 - P\Sigma P + \Upsilon^T P\Sigma P\Upsilon \quad (3.8)$$

$$0 = (A - \Sigma P)\hat{Q} + \hat{Q}(A - \Sigma P)^T + Q\overline{\Sigma}Q - \Upsilon Q\overline{\Sigma}\Upsilon^T \quad (3.9)$$

$$0 = (A - Q\overline{\Sigma})^T\hat{P} + \hat{P}(A - Q\overline{\Sigma}) + P\Sigma P - \Upsilon^T P\Sigma P\Upsilon \quad (3.10)$$

The solution of the above equations is obviously not a trivial task. The solution method proposed by Hyland [52] is based on a parameter optimisation procedure which, again, is not particularly attractive from a computational viewpoint, although it has been claimed in [51] that the computational requirements of obtaining a low order controller using this method are lower than the computational requirements of designing a full order optimal controller and then obtaining a low order controller by using the method of Modal Cost Analysis [36].

The coupling of equations 3.7 to 3.10 highlights an important point concerning the design of low order controllers for high order systems, in that it indicates the demise of the well-known "Separation Theorem" [53],[54, chapter 6]. The Separation Theorem states that a state estimator and a state feedback law can be designed for a system independently of each other, that is the design of the state estimator will not cause any poles to shift from their locations specified by the state feedback law, or vice versa. However, the coupling terms in equations 3.7 to 3.10 clearly indicate that if a reduced order state feedback law and a reduced order state estimator is to be designed for a system, (or if there are unmodelled dynamics in the system) then the Separation Theorem is no longer valid. This is indeed confirmed in Chapter 4.

3.5 Summary

Three approaches for designing low order controllers for high order systems have been considered. The first approach examined was based on reducing the order of the model to a sufficiently low level such that a controller designed for this low order model could be implemented. Several methods of computing the low order model have been discussed and, in particular, the method of Modal Cost Analysis appears to be the most useful, both in terms of its low computational requirements, and its ease of application to very high order systems.

However, deriving a low order model to represent a high order model (which itself is only approximate) as a means of designing a feedback system is a rather undesirable approach, due to the generally unpredictable effects of the feedback law on the unmodelled dynamics. Controllers can generally be designed for reduced order models which result in a stable closed loop system when the controller is applied to the reduced order model (assuming, of course, that the reduced order model is stabilisable), however when the same controller is applied to the original high order model, there may be a significant decrease in performance of the closed loop system, and in some cases, the closed loop system may even be unstable. This problem is examined in greater detail in Chapter 4.

The second approach examined attempts to avoid this problem by designing a controller for the high order model and then reducing the order of the controller to a level which could be implemented, but maintaining the essential characteristics of the controller. Two methods for reducing the order of the controllers have been examined, the first based on matching the Markov parameters and output covariances of the closed loop system, (the "Q-cover" algorithm), and the second method based on retaining the states whose feedback contribute most to the performance of the closed loop system as defined by the cost function, (the Modal Cost Analysis method).

A disadvantage of the Q-cover algorithm is that it does not guarantee a stable closed loop system with the reduced order controller, thus it may be necessary to "search" for a stabilising reduced order controller of a particular order by adjusting the initial conditions of the algorithm. In addition, computation of the powers of matrix A_c in the construction of the observability matrix Θ_q can cause numerical problems, particularly with numerically "stiff" systems.

The Modal Cost Analysis method of controller reduction is very similar to the Modal Cost Analysis method of model reduction, and thus the

computational requirements are fairly simple. However, the method only reduces the state feedback gain matrix, and the problem of designing a state estimator to estimate the required states still remains. As will be seen in the next chapter, it is not a trivial task to design a state estimator of low order for a high order system which does not corrupt the locations of the system poles.

Of course, a problem for any method which requires the design of a controller for a high order system, (which may, or may not, be subsequently reduced in order), is the computational difficulties of the design of the high order controller. In addition to high computational cost, many controller design methods are susceptical to numerical difficulties when applied to high order systems, and thus the choice of design method may be severely restricted.

The third approach examined considers the design of a controller for a high order system which incorporates constraints in the design method such that the resultant controller will be of sufficiently low order that it may be implemented. One such method which has been discussed is known as the "Optimally Projected Equations" method. Although this method is attractive because no approximations of the model or the controller are necessary, it does have one serious drawback. The low order controller is obtained by simultaneously solving a set of four coupled equations, in particular, a pair of modified Riccati equations, and a pair of modified Lyapunov equations. This is obviously not a trivial problem, and for this reason, this method has not been examined further. However, these equations highlight an important point regarding the well-known Separation Theorem which states that a state feedback law and a state estimator can be designed for a system completely independently of each other. In the presence of unmodelled dynamics however, (or when designing a controller using a reduced order model), the theorem no longer holds true, and the interaction between the state feedback law and the state estimator gives rise to the coupling terms in equations 3.7 to 3.10. This is confirmed in the next chapter.

An alternative approach to the design of low order controllers for a high order system is to constrain the controller to use an output feedback law directly, rather than using the outputs to assist in the reconstruction of the system states for use by a state feedback law, as generally the number of outputs of a system will be much lower than the number of states. Direct output feedback also has certain useful stability properties, (which are discussed in the next chapter), but it also has inherently lower performance capabilities, in general, than state feedback. Under certain constraints, output feedback

laws can be designed using a very low order model, (in fact, by using a model which completely ignores the elastic behaviour altogether) and the stability of the closed loop system can be guaranteed, but the transient behaviour of the elastic modes cannot be specified.

In practise, the design of a controller for a large flexible space structure will probably require a combination of approaches. For example, even if it is desired to use the "high order design – controller reduction" approach, it is very likely that the numerical difficulties associated with the design of high order controllers will necessitate the reduction of the open loop model to a level such that the high order controller design can be carried out, and the controller reduction method may then be applied in order to obtain the implementable controller.

Chapter 4

Application of Feedback to Flexible Structures

4.1 Introduction

In this chapter the effects of applying feedback to flexible structures are examined from an analytical viewpoint. Initially, the effects on stability of using the outputs of the system for feedback via fixed gains are examined, then the effects of using the states of the system for feedback via fixed gains are examined. When the states are not available for feedback, some form of state estimator is required to reconstruct the states and the subsequent effects of this are also examined.

Because a finite dimension state space model (or equally, a transfer function matrix model) is used to represent an infinite dimension system, it is inherent that a reduced order model is used for feedback law design, (or alternatively, that there are unmodelled dynamics in the system), so the implications of this are examined by considering the system to be partitioned into "retained" and "neglected" subsystems.

The dynamic state estimators considered in later sections of this chapter are by no means a complete set, but merely three types of dynamic state estimator that are commonly used for linear multivariable systems. A more exhaustive study of dynamic state estimators for linear systems can be found in the text by O'Reilly [55].

The structure to be controlled is to be represented by a state space model (see Chapter 2), where the state vector x, and the output vector y

are defined as;-

$$x_i(t) = \begin{pmatrix} q_i(t) \\ \dot{q}_i(t) \end{pmatrix}$$
$$y_i(t) = \begin{pmatrix} d_i(t) \\ \dot{d}_i(t) \end{pmatrix}$$

where q_i is the *ith* modal co-ordinate, and d_i is the *ith* nodal co-ordinate (degree of freedom). The state space model is of the form;-

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \quad (a) \\ y(t) &= Cx(t) \quad (b) \end{aligned} \tag{4.1}$$

where matrix A is block-diagonal with blocks of the form;-

$$a_i = \begin{bmatrix} 0 & 1 \\ -\omega_i^2 & -2\zeta_i\omega_i \end{bmatrix}$$

and matrix B has blocks of the form;-

$$b_i = \begin{bmatrix} 0 \\ \phi_1^T \end{bmatrix}$$

where ϕ_1 is the vector describing the shape of mode i at the locations of the actuators, and matrix C has blocks of the form;-

$$c_i = \begin{bmatrix} \phi_2 & 0 \\ 0 & \phi_2 \end{bmatrix}$$

where ϕ_2 is the vector describing the shape of mode i at the locations of the sensors.

The effects of feedback with the ignored dynamics alone are to be considered here, so it will be assumed that the system is completely noise-free.

4.2 Output Feedback

Consider the above model to be subject to output feedback, that is a feedback law of the form;-

$$u(t) = -Fy(t) \tag{4.2}$$

The resulting closed loop system is given by;-

$$\begin{aligned} \dot{x}(t) &= (A - BFC)x(t) \\ y(t) &= Cx(t) \end{aligned}$$

and obviously it is desirable to select the matrix F such that the matrix $(A - BFC)$ is stable, which is possible if the matrix triple (A, B, C) is stabilisable.

However, because the order of a model of a flexible structure should in theory be infinite, a reduced order model is implicit in the design of feedback laws for such systems, and this is now examined.

Consider that the model given by equations 4.1 is now partitioned such that x_r represents the states that are retained in a model which will be used for feedback law design, and x_n represents the remaining states, which are neglected. Thus the partitioned system can now be described as follows;-

$$\begin{aligned} \begin{pmatrix} \dot{x}_r(t) \\ \dot{x}_n(t) \end{pmatrix} &= \begin{bmatrix} A_r & 0 \\ 0 & A_n \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \end{pmatrix} + \begin{bmatrix} B_r \\ B_n \end{bmatrix} u(t) \quad (a) \\ y(t) &= \begin{bmatrix} C_r & C_n \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \end{pmatrix} \qquad (b) \end{aligned} \tag{4.3}$$

Note that the partitioned A matrix is block-diagonal because the original A matrix itself is block-diagonal. The reduced order model used for feedback law design is given by;-

$$\begin{aligned} \dot{x}_r(t) &= A_r x_r(t) + B_r u(t) \quad (a) \\ y(t) &= C_r x_r(t) \qquad (b) \end{aligned} \tag{4.4}$$

The feedback law (equation 4.2) is then defined for the system given by equations 4.4 such that the closed loop system becomes;-

$$\begin{aligned} \dot{x}_r(t) &= (A_r - B_r F C_r)\, x_r(t) \\ y(t) &= C_r x_r(t) \end{aligned}$$

However, the effect of using a reduced order model for defining the feedback law can now be examined by applying the feedback law to the complete system, defined by equations 4.3, which results in the following closed loop

system;-

$$\begin{pmatrix} \dot{x}_r(t) \\ \dot{x}_n(t) \end{pmatrix} = \begin{bmatrix} (A_r - B_r F C_r) & -B_r F C_n \\ -B_n F C_r & (A_n - B_n F C_n) \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \end{pmatrix} \quad (a)$$

$$y(t) = \begin{bmatrix} C_r & C_n \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \end{pmatrix} \quad (b)$$

(4.5)

It is obvious from equations 4.5 that the closed loop poles of the complete system are not given by the poles of the design model, and the poles of the ignored part of the system, but are given by an interaction of the two "subsystems".

It is now possible to identify three popular terms that have been adopted by many authors, these being "spillover", "control spillover", and "observation spillover". The term "spillover" refers to the general problem of unpredicted pole shifts due to the modelling errors implicit in the use of a reduced order model, "control spillover" refers to the unpredicted pole shifts particularly connected with the ignored input terms (that is the terms associated with B_n in the closed loop system matrix), and "observation spillover" refers to the unpredicted pole shifts particularly connected with the ignored output terms (that is the terms associated with C_n in the closed loop system matrix). Thus, in equation 4.5(a) the term $-B_n F C_r$ represents "control spillover", the term $-B_r F C_n$ represents "observation spillover", and the term $-B_n F C_n$ represents both "control spillover" and "observation spillover".

Although initially output feedback may appear most unsuitable for use on flexible structures, it has the advantage of being simple to implement, and consequently it has been the subject of much work. Several authors have examined the problems of "spillover" and have been able to predict closed loop stability for flexible structures subject to output feedback involving certain constraints on the feedback gains, and these results are discussed in section 4.4.1.

4.3 State Feedback

4.3.1 Perfect States

Consider initially that the states are available for feedback, and thus it is possible to construct a feedback law of the form;-

$$u(t) = -Fx(t)$$

If this is applied to the system defined by equations 4.1 the resulting closed loop system is given by;-

$$\begin{aligned} \dot{x}(t) &= (A - BF)x(t) \\ y(t) &= Cx(t) \end{aligned}$$

Again, it is obviously desirable to choose matrix F such that the matrix $(A-BF)$ is stable, which can be done if the matrix pair (A, B) is stabilisable. As in the previous section, the effect of using a reduced order model for designing the feedback law is examined. Hence consider the partitioned system defined by equations 4.3 subject to a control law of the form;-

$$u(t) = -Fx_r(t)$$

The resulting closed loop system is given by;-

$$\begin{aligned} \begin{pmatrix} \dot{x}_r(t) \\ \dot{x}_n(t) \end{pmatrix} &= \begin{bmatrix} (A_r - B_rF) & 0 \\ -B_nF & A_n \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \end{pmatrix} \quad (a) \\ y(t) &= \begin{bmatrix} C_r & C_n \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \end{pmatrix} \quad (b) \end{aligned} \tag{4.6}$$

As the closed loop matrix in equation 4.6(a) is block lower triangular, the poles of the system are given by the poles of the diagonal blocks, $(A_r - B_rF)$ and A_n, thus there are no unpredicted pole shifts. It can be seen from equation 4.6(a) that a "control spillover" term $-B_nF$ is present, but no "observation spillover" term is present. The lack of an "observation spillover" term is due to the assumption that the states x_r are available with complete accuracy. This result led to the statement by Balas [56] that a sufficient condition to guarantee closed loop stability of a flexible structure subject to a feedback law designed using a reduced order model was that either "control spillover" or "observation spillover" should be zero. However,

as will be become apparent, this condition is very difficult, if not impossible, to achieve with any practical control system design.

Although these results suggest that state feedback is very attractive, they have been based on the assumption that the required states are available with complete accuracy, but in the case of flexible structures the states are generally not physical variables so it is impossible to measure them, hence this assumption is not valid. In the next section, the effects of having to estimate the states on closed loop stability are examined.

4.3.2 Estimated States

4.3.2.1 Static State Estimators

Now consider the more realistic case where the states are not directly available and so have to be reconstructed in some manner. Initially, consider the use of a static state estimator, that is for a system given by;-

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned}$$

where $x \in \mathcal{R}^n$ and $y \in \mathcal{R}^p$, then the states may be reconstructed directly from the outputs by a static estimator of the form;-

$$\hat{x}(t) = Ky(t)$$

where $\hat{x}$ is the estimate of the system state x, and matrix K is such that;-

$$KC = I$$

where I is the unit matrix of dimension n.

If matrix C is non-square, as in general it will be, then matrix K can be obtained as the pseudo-inverse of C, (sometimes known as the Moore-Penrose inverse, or the generalised inverse) such that;-

$$K = C^T \left(CC^T\right)^{-1}$$

If C is square, and non-singular, (that is the number of outputs equals the number of states, and all the outputs are independent) then this reduces to the more usual inverse such that;-

$$K = C^{-1}$$

The fundamental drawback of using a static state estimator is that it assumes an accurate knowledge of the output matrix (the C matrix) of the system. Models of flexible structures are generally accepted as not being particularly well-known, and indeed, their representation by a finite order model is an approximation in itself. Thus the use of a static state estimator could be expected to give rise to robustness problems, in the sense of being rather sensitive to modelling errors and parameter variations.

A feedback law design method which incorporates a static state estimator has been proposed for flexible spacecraft by Meirovitch [4], and this is examined further in Chapter 5.

4.3.2.2 Dynamic State Estimators

Now consider a general dynamic state estimator which attempts to reduce the sensitivity problems of static state estimators by using more information from the system. It should be noted that a dynamic state estimator still incorporates a model of the open loop system within its structure, although the model used is more complete than just modelling the output matrix as in the static state estimator.

Consider again the system defined by equations 4.3 but now subject to a feedback law of the form;-

$$u(t) = -F\hat{x}_r(t) \tag{4.7}$$

where $\hat{x}_r(t)$ is an estimate of $x_r(t)$ obtained via a dynamic state estimator of the form;-

$$\begin{aligned} \dot{z}(t) &= Dz(t) + Gu(t) + Hy(t) \quad (a) \\ \hat{x}_r(t) &= K_1 y(t) + K_2 z(t) \qquad\quad (b) \end{aligned} \tag{4.8}$$

Before the effects on closed loop stability of this system are examined, the closed loop system comprised of the design model subject to a state feedback law obtained via a dynamic state estimator will be examined in order to identify certain useful design constraints for the selection of the matrices in the estimator. (For a thorough treatment, see [53],[57].) Thus consider the system given by;-

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned}$$

subject to a feedback law of the form;-

$$u(t) = -F\hat{x}(t)$$

where $\hat{x}(t)$ is an estimate of $x(t)$ obtained via a dynamic state estimator of the form;-

$$\begin{aligned} \dot{z}(t) &= Dz(t) + Gu(t) + Hy(t) \quad (a) \\ \hat{x}(t) &= K_1 y(t) + K_2 z(t) \quad (b) \end{aligned} \tag{4.9}$$

This results in the closed loop system given by;-

$$\begin{pmatrix} \dot{x}(t) \\ \dot{z}(t) \end{pmatrix} = \begin{bmatrix} (A - BFK_1C) & -BFK_2 \\ (HC - GFK_1C) & (D - GFK_2) \end{bmatrix} \begin{pmatrix} x(t) \\ z(t) \end{pmatrix}$$

If an error function $e(t)$ is now defined such that;-

$$e(t) = z(t) - Lx(t)$$

it is now possible to define the closed loop system in terms of $x(t)$ and $e(t)$, as follows;-

$$\begin{pmatrix} \dot{x}(t) \\ \dot{e}(t) \end{pmatrix} = \begin{bmatrix} (A - BFP) & -BFK_2 \\ (QFP + DL + HC - LA) & (QFK_2 + D) \end{bmatrix} \begin{pmatrix} x(t) \\ e(t) \end{pmatrix} \tag{4.10}$$

where the matrices P and Q are defined as;-

$$\begin{aligned} P &= K_1C + K_2L \\ Q &= LB - G \end{aligned}$$

From equations 4.10 it is obviously desirable to make;-

$$K_1C + K_2L = I \tag{4.11}$$

and if matrix G is selected such that;-

$$G = LB \tag{4.12}$$

and also H and L are chosen such that;-

$$LA - DL - HC = 0 \tag{4.13}$$

then the closed loop system defined by equations 4.10 becomes;-

$$\begin{bmatrix} \dot{x}(t) \\ \dot{e}(t) \end{bmatrix} = \begin{bmatrix} (A - BF) & -BFK_2 \\ 0 & D \end{bmatrix} \begin{bmatrix} x(t) \\ e(t) \end{bmatrix}$$

Thus the poles of the system are given by the poles of the blocks $(A - BF)$ and D. Hence the poles of the "plant" are positioned as if the exact states were available for feedback, and the matrix D can be selected to be a stable matrix with poles several times faster than the poles of A so that the estimates of the states converge rapidly to the correct values. The design of dynamic state estimators is examined in detail in Chapter 5.

Returning to the initial problem defined by equations 4.3, 4.7, and 4.8, the closed loop system is given by;-

$$\begin{pmatrix} \dot{x}_r(t) \\ \dot{x}_n(t) \\ \dot{z}(t) \end{pmatrix} = \begin{bmatrix} (A_r - B_rFK_1C_r) & -B_rFK_1C_n & -B_rFK_2 \\ -B_nFK_1C_r & (A_n - B_nFK_1C_n) & -B_nFK_2 \\ (HC_r - GFK_1C_r) & (HC_n - GFK_1C_n) & (D - GFK_2) \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \\ z(t) \end{pmatrix} \tag{4.14}$$

Following the earlier analysis, if an error function $e(t)$ is now defined such that;-

$$e(t) = z(t) - Lx_r(t) \tag{4.15}$$

it is now possible to define the closed loop system in terms of $x_r(t)$, $x_n(t)$, and $e(t)$, and with the appropriate changes to equations 4.11 to 4.13 as follows;-

$$K_1C_r + K_2L = I \tag{4.16}$$

$$G = LB_r \tag{4.17}$$

$$LA_r - DL - HC_r = 0 \tag{4.18}$$

the closed loop system is given by;-

$$\begin{pmatrix} \dot{x}_r(t) \\ \dot{x}_n(t) \\ \dot{e}(t) \end{pmatrix} = \begin{bmatrix} (A_r - B_rF) & -B_rFK_1C_n & -B_rFK_2 \\ -B_nF & (A_n - B_nFK_1C_n) & -B_nFK_2 \\ 0 & HC_n & D \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \\ e(t) \end{pmatrix} \tag{4.19}$$

It is obvious from equation 4.19 that the poles of the closed loop system no longer correspond to the poles given by the blocks on the diagonal, and thus the "spillover" problem associated with output feedback laws as

demonstrated in section 4.2 is just as prevalent with state feedback laws in the practical case where a dynamic state estimator is involved. This is not particularly surprising, because a dynamic state estimator uses information from the system outputs to help reconstruct the estimates of the system states.

This general analysis is now extended to examine particular implementations of dynamic state estimators.

4.3.2.2.1 Minimal Order Observer This type of dynamic state estimator is accredited to the work of Luenberger [53],[57], and is known as a "minimal order observer" or as a "Luenberger observer". In particular, Luenberger showed that the minimum order necessary for a dynamic state estimator is l, where $l = n - p$, and n is the number of states of the system, and p is the number of independent outputs of the system. The form of this type of observer is exactly the same as the general dynamic state estimator discussed in the previous section. Hence it is now possible to consider the dimensions of the various matrices which define the minimal order observer. Thus from equations 4.8,4.16, and 4.17, the minimal order observer is given by;-

$$\dot{z}(t) = Dz(t) + LB_r u(t) + Hy(t) \tag{4.20}$$

$$\hat{x}_r(t) = K_1 y(t) + K_2 z(t) \tag{4.21}$$

where;-

$$LA_r - DL - HC_r = 0 \tag{4.22}$$

where $x_r \in \mathcal{R}^n$, $u \in \mathcal{R}^m$, $y \in \mathcal{R}^p$, $\hat{x}_r \in \mathcal{R}^n$, and $z \in \mathcal{R}^l$, $(l = n - p)$.

The dimensions of the matrices are inferred from the dimensions of the vectors. (See figure 4.1 for a block diagram of the structure of a minimal order observer.)

Obviously, the equations describing the closed loop system are identical to those given by equations 4.14 or 4.19, and thus the non-zero off-diagonal blocks suggest that the closed loop stability of the complete system cannot be implied by stability of the component parts.

4.3.2.2.2 Full Order Observer The "full order observer" is so-called because the dimension of the estimator state vector is the same as the dimension of the system state vector that the estimator is attempting to reconstruct. This type of dynamic state estimator is sometimes referred to as

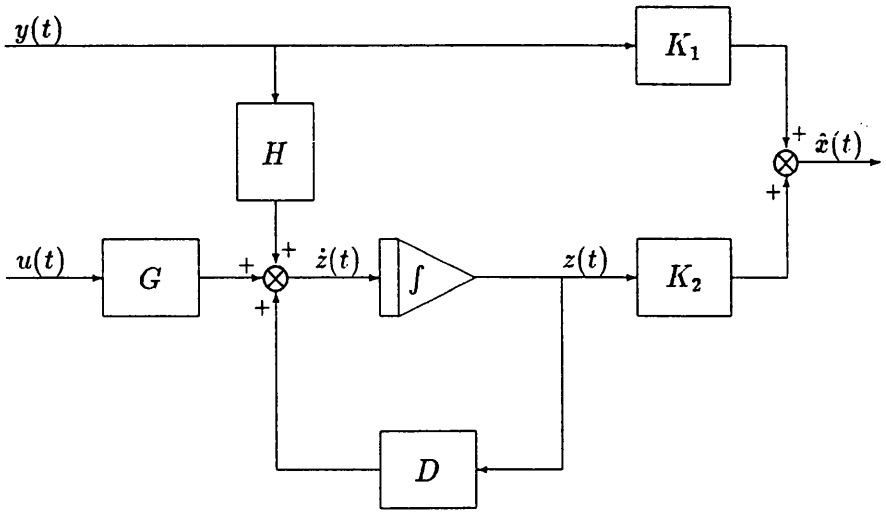


Figure 4.1: Block diagram of minimal order observer

an "asymptotic observer", although strictly speaking, the term "asymptotic observer" refers to a *class* of dynamic state estimators, to which all of the dynamic state estimators discussed in this chapter belong. For more details, see O'Reilly [55].

Recalling equations 4.15, if the matrix L is set to unity, that is $L = I$, then equations 4.16 to 4.18 become;-

$$K_1 C_r + K_2 = I \tag{4.23}$$

$$G = B_r \tag{4.24}$$

$$A_r - D - HC_r = 0 \tag{4.25}$$

Equation 4.23 can be conveniently satisfied by setting $K_2 = I$, and K_1=0, and thus equations 4.20 and 4.21 become;-

$$\dot{z}(t) = (A_r - HC_r)\, z(t) + B_r u(t) + Hy(t) \tag{4.26}$$

$$\hat{x}_r(t) = z(t) \tag{4.27}$$

where $x_r \in \mathcal{R}^n$, $u \in \mathcal{R}^m$, $y \in \mathcal{R}^p$, $\hat{x}_r \in \mathcal{R}^n$, and $z \in \mathcal{R}^n$. (See figure 4.2 for a block diagram of the structure of a full order observer.)

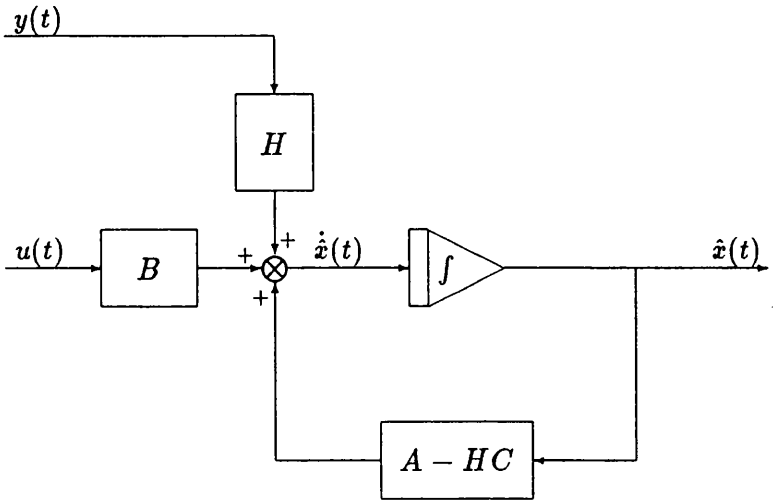


Figure 4.2: Block diagram of full order observer

Re-arranging equation 4.26 and incorporating equation 4.27 gives the following form;-

$$\dot{\hat{x}}_r(t) = A_r\hat{x}_r(t) + B_r u(t) + H\left(y(t) - C_r\hat{x}_r(t)\right)$$

This equation shows more clearly the model of the system within the estimator. The model incorporated into the minimal order observer is in a projected basis and so is identified less readily than the model in the full order observer which is in the same basis as the system.

The equations which describe the behaviour of the closed loop system can be derived from equations 4.14, 4.23, and 4.24 using the conditions $L = I$, $K_1 = 0$, and $K_2 = I$, as follows;-

$$\begin{pmatrix} \dot{x}_r(t) \\ \dot{x}_n(t) \\ \dot{z}(t) \end{pmatrix} = \begin{bmatrix} A_r & 0 & -B_rF \\ 0 & A_n & -B_nF \\ HC_r & HC_n & (A_r - HC_r - B_rF) \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \\ z(t) \end{pmatrix} \quad (4.28)$$

where $z(t) = \hat{x}_r(t)$. With the error function $e(t)$ now defined from equation 4.15 as;-

$$e(t) = \hat{x}_r(t) - x_r(t)$$

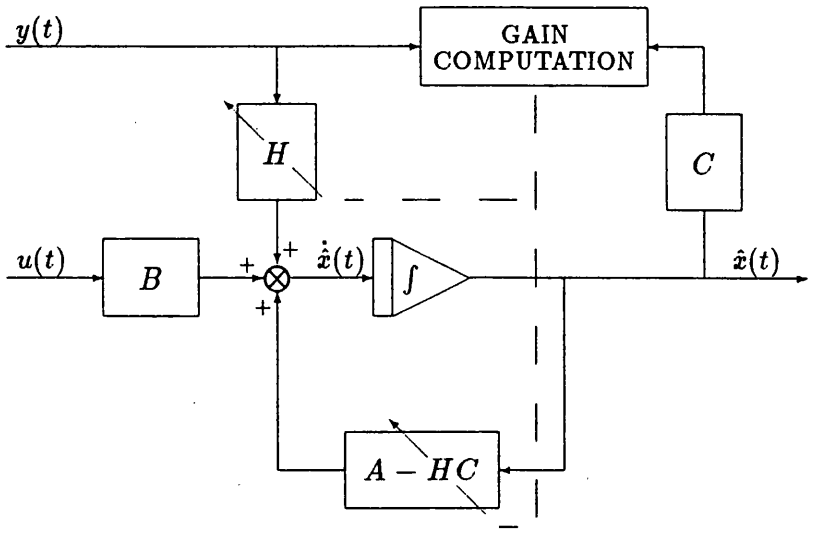


Figure 4.3: Block diagram of Kalman Filter

an alternative description of the closed loop system is;-

$$\begin{pmatrix} \dot{x}_r(t) \\ \dot{x}_n(t) \\ \dot{e}(t) \end{pmatrix} = \begin{bmatrix} (A_r - B_r F) & 0 & -B_r F \\ -B_n F & A_n & -B_n F \\ 0 & HC_n & (A_r - HC_r) \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \\ e(t) \end{pmatrix}$$

Again it can be seen that the "spillover" problem is very apparent, the non-zero off-diagonal blocks suggesting that closed loop stability of the complete system cannot be implied by stability of the component systems.

4.3.2.2.3 Kalman Filter This type of dynamic state estimator is accredited primarily to the work of Kalman [58]. The structure of the Kalman Filter is essentially the same as that of the full order observer but its unique aspect is the method in which the gain matrix H is chosen, which attempts to take into account the noise in the system and the noise in the measurements. (See block diagram in figure 4.3.)

Consider the system described by the state space equations;-

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) + w(t) \\ y(t) &= Cx(t) + v(t) \end{aligned}$$

where $w(t)$ is a vector representing the system noise, and $v(t)$ is a vector representing the measurement noise.

The system noise term $w(t)$ is assumed to be a set of independent, zero mean, white noise sources, having a covariance matrix Q, where Q is diagonal and each element q_i is the power spectral density of the noise source $w_i(t)$. Similarly, the measurement noise $v(t)$ is assumed to be a set of independent, zero mean, white noise sources having an associated covariance matrix R, where R is diagonal and each element r_i is the power spectral density of the noise source $v_i(t)$.

The full derivation of the equations defining the Kalman Filter can be found in [59], so the derivation is not repeated here, but the equations are merely stated in the form;-

$$\dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t) + H(t)\left(y(t) - C\hat{x}(t)\right)$$

where;-

$$H(t) = P(t)C^T R^{-1}$$

and

$$\dot{P}(t) = AP(t) + P(t)A^T + Q - P(t)C^T R^{-1} C P(t) \qquad (4.29)$$

The matrix $P(t)$ is the covariance matrix of the estimates $\hat{x}(t)$, and thus the Kalman Filter attempts to choose the gain matrix $H(t)$ in such a manner as to minimise the covariance of the estimates $\hat{x}(t)$. Note that equation 4.29 is known as the Matrix Riccati Equation. (The steady-state equation, that is when $\dot{P}(t) = 0$, is known as the Algebraic Matrix Riccati Equation.)

Obviously, the effects on closed loop stability of a system where the Kalman Filter and state feedback law are designed using a reduced order model will be the same as for the use of a full order observer and state feedback, as discussed in the previous section, the only difference being that the estimator gain matrix H will be time varying, that is $H(t)$.

Although the noise terms used in the derivation of the Kalman Filter were assumed to be zero mean, white noise sources, consider the case when a reduced order model is used for feedback law design. As was shown in the previous section, the neglected part of the output matrix (the C matrix) gives rise to "observation spillover" terms, thus a state estimator which provided some rejection of the ignored output components could be expected to perform better than a standard state estimator, such as the full order observer discussed in the previous section.

If the ignored output components happened to be zero mean, white noise signals, then the Kalman Filter would provide "optimal" rejection of these

components. However, the ignored components are not zero mean, white noise signals, for example the "output noise" is given by;-

$$v(t) = C_n x_n$$

where;-

$$\dot{x}_n = A_n x_n + B_n u$$

Obviously the "noise" term will not be uncorrelated to the system states (linked by the term $B_n u$ and the feedback law $u = -Fx_r$), and the "noise" will not in general have a uniform frequency distribution. However, as long as the closed loop system is stable, the states x_n will have zero mean (recall that they represent vibration terms) and will tend to zero, and because the unmodelled modes will in theory cover an infinite bandwidth, it may be conjectured that a Kalman Filter would provide some rejection of these terms, although certainly not an "optimal" rejection.

Another way in which the theory of the Kalman Filter may be useful is in the guidance of the choice of gains for a fixed gain dynamic state estimator. It is well-known that if the ratio of system noise to measurement noise is small, then the Kalman Filter gains will tend to be small, thus the estimates will depend more heavily on the internal model of the system within the Filter rather than on the measurements of the system. Conversely, if the ratio of the system noise to measurement noise is large, then the Kalman Filter gains will tend to be large, thus the estimates will depend more heavily on the measurements of the system rather than on the internal model within the Filter.

In the case of flexible space structures, the system model is not generally known particularly accurately, and the outputs are "corrupted" with a "noise" term. The inaccurate knowledge of the system model suggests the use of a high gain in a dynamic state estimator, but this is opposed by the output "noise" which suggests the use of a low gain in an estimator. Hence the selection of gains for a dynamic state estimator for flexible structures needs to be a compromise between the two requirements.

Indeed, it may be easier to design a reliable high order, fixed gain dynamic state estimator by using the steady-state solution to the Kalman Filter problem which may be computed off-line allowing the adjustment of the solution by choice of the noise term weightings, rather than using a pole placement type of approach where it is assumed that there is no noise in the system or measurements, and where little guidance is available as to the "best" locations for the poles. This is discussed in more detail in Chapter 5.

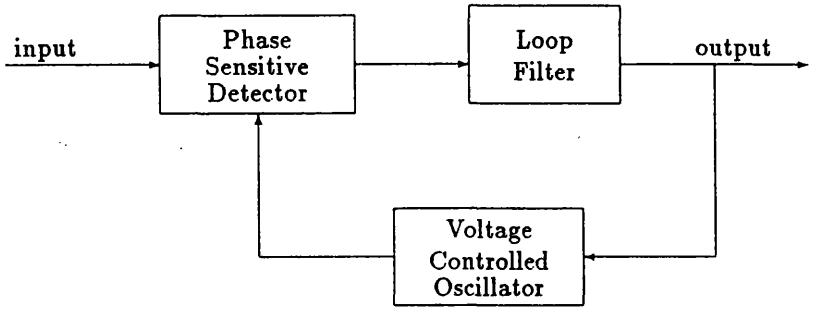


Figure 4.4: Block diagram of basic Phase-locked Loop

4.3.2.2.4 Other Schemes Some less conventional methods have been proposed in the literature for the dynamic estimation of states of flexible space structures, and two such methods are briefly discussed here.

The first method to be discussed is based on phase-locked loops [60]. A basic phase-locked loop is composed of a voltage controlled oscillator, a phase sensitive detector, and a filter. (See figure 4.4.) An analysis of a phase-locked loop will not be given here as it is readily available in many texts on communications systems (for example, see [61]). A simple description of its behaviour is that the loop attempts to control the voltage controlled oscillator such that the frequency of the oscillator matches the frequency of the input signal. With some additional hardware, a phase-locked loop can be used as the basis of a tunable bandpass filter, thus a particular frequency component of an input signal could be extracted. Recall that the states of the system correspond to vibration at particular frequencies, hence it may be possible to isolate, and thus identify, the states by frequency.

However, state estimation for flexible space structures using phase-locked loops is not particularly attractive because of the possibility of closely grouped frequencies, which would cause difficulties with loop acquisition, in that it would be extremely difficult to guarantee which phase-locked loop would lock onto which frequency, and mutual exclusion would also be difficult to guarantee. Also, as the structural transients decayed, the input signal amplitudes would drop which would cause the phase-locked loops to lose acquisition, so a certain minimum level of excitation may be necessary for the loops to maintain acquisition.

The second method to be discussed is based on least squares lattice filters. The lattice filter, together with a discrete Fourier transform, is used to provide on-line estimates of the number of significant modes, and their shapes, and the corresponding modal amplitude time series [62].

The performance of this type of estimator as part of an adaptive control scheme has been examined by a simulation study of its application to a 12ft "free-free" beam [63], where it was shown to perform quite well.

However, a drawback of this method is the fact that the number of modes to be controlled is not known *a priori* and hence it may be necessary to over-specify the requirements of the control processor in order to allow sufficient processing time should the identified number of modes exceed the expected number, which may be undesirable.

4.4 Conditions for Closed Loop Stability

4.4.1 Systems with Output Feedback

Balas [64] showed using Lyapunov stability theory that for a system where an equal number of collocated force (or torque) actuators and velocity (or angular velocity) sensors are used, and where the actuators do not excite any zero frequency modes, if they exist, then provided that the feedback matrix F is non-negative definite, the closed loop system will be stable, in the sense of Lyapunov, irrespective of the choice of feedback gains used in matrix F.

These constraints mean that these results cannot be applied to a controller designed to control the rigid modes of a structure, but they may be used as part of an overall control scheme, where a velocity feedback system is used to augment the damping of a structure over a wide bandwidth, whilst a more complex type of controller provides the means of achieving "pointing" control. This is discussed in more detail in the next chapter.

A similar result has also been obtained by Arbel and Gupta [65] by considering an inverse optimal control problem. (A type of problem where the control is known, and it is desired to obtain the weighting matrices which minimise the cost function.) For the same actuator/sensor conditions used by Balas, as discussed above, Arbel and Gupta showed that if the feedback matrix F is positive definite, then asymptotic stability of the closed loop system is guaranteed. This result was extended such that the authors proposed an iterative method for the computation of the feedback gains, where termination of the algorithm was determined by "engineering judgement"

rather than numerical convergence of some kind.

An alternative use for the results of Balas, and Arbel and Gupta is for the design of "member damper" controllers. This type of controller is effectively an active "dashpot", such that between the two points to which a "member damper" is connected, equal and opposite forces proportional to the sensed relative velocity between the points, are applied. (A dual of the force damper has also been considered, where equal and opposite torques are exerted proportional to the sensed relative angular velocity between the two points.) The use of "member dampers" have been proposed for use in complex truss-like structures [66] where the damping of the overall structure can be increased by the use of local feedback.

The equations describing "member damper" control are detailed, using the modal equations defined in the previous chapter, as follows;-

$$\begin{aligned} \ddot{q}_i(t) + 2\zeta_i\omega_i\dot{q}_i(t) + \omega_i^2 q_i(t) &= \begin{pmatrix} \phi_{i1}^T & \phi_{i2}^T \end{pmatrix} \begin{pmatrix} f_i(t) & -f_i(t) \end{pmatrix}^T \\ &= \left(\phi_{i1}^T - \phi_{i2}^T\right) f_i(t) \qquad (4.30) \\ \dot{y}_i(t) &= (\phi_{i1} - \phi_{i2})\,\dot{q}_i(t) \qquad (4.31) \\ f_i &= -F\dot{y}_i \qquad i = 1,\ldots n \qquad (4.32) \end{aligned}$$

Substituting;-

$$\psi_i = (\phi_{i1} - \phi_{i2})$$

equations 4.30 to 4.32 become;-

$$\begin{aligned} \ddot{q}_i(t) + 2\zeta_i\omega_i\dot{q}_i(t) + \omega_i^2 q_i(t) &= \psi_i^T f_i(t) \\ \dot{y}_i(t) &= \psi_i\dot{q}_i(t) \\ f_i(t) &= -F\dot{y}_i(t) \qquad i = 1,\ldots n \end{aligned}$$

which is identical to the usual rate output feedback case, and where the closed loop equations are;-

$$\ddot{q}_i(t) + \left(2\zeta_i\omega_i - \psi_i^T F\psi_i\right)\dot{q}_i(t) + \omega_i^2 q_i(t) = 0 \quad i = 1,\ldots n$$

thus demonstrating how direct velocity feedback augments the natural damping of a structure.

It must be noted that the results obtained by Balas, and Arbel and Gupta are only sufficient conditions for stability, not necessary conditions.

More recent work by Joshi [67] has extended the stability results for direct velocity feedback to include limited actuator and sensor dynamics, and

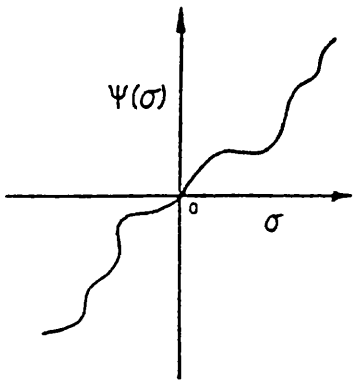


Figure 4.5: An example of a non-linearity in the $(0, \infty)$ sector

certain non-linearities in the feedback loops. In particular, for a velocity feedback gain matrix F which is positive-definite, then asymptotic stability is still guaranteed in the presence of linear time-invariant sensor and actuator dynamics provided that the phase angle corresponding to such dynamics is within $\pm 90°$. This condition is, of course, satisfied for stable first order dynamics. In addition, asymptotic stability is also guaranteed in the presence of time-varying or time-invariant non-linearities in the feedback loops, provided that the non-linearities belong to the $(0, \infty)$ sector, that is, they lie in the 1st and 3rd quadrants, see figure 4.5.

West-Vukovich, Davison, and Hughes [44] obtained some interesting results by considering "decentralised control", where sensor signals at any location are used exclusively as feedback signals for actuators at the same location, (and that the actuators and sensors at any location are "dual", that is an angular displacement and/or rate sensor combined with a torque actuator, or a linear displacement and/or rate sensor with a force actuator), that is collocated, mutually dual actuators and sensors with diagonal feedback gain matrices. The result was proved that irrespective of the flexibility parameters, the closed loop system was guaranteed to be asymptotically stable provided that the rate feedback gain matrix and the position feedback gain matrix were both positive-definite.

An alternative proof derived by the author, and some discussion of the implications of these results can be found in Appendix A.2.

Note that all the above results have assumed that the open loop system is stabilisable. This implies that any poles of the open loop system which are uncontrollable and unobservable (sometimes referred to as the "fixed modes" of the system [44],[45]) are stable. For all practical cases of flexible space structures this will be true, as the natural damping, even though small, will ensure that such "fixed modes" are stable, as long as the set of "fixed modes" does not include any rigid modes.

4.4.2 Systems with Estimated State Feedback

Some conditions have been obtained by Joshi and Groom [66] based on Lyapunov methods which are sufficient to ensure asymptotic stability of the closed loop system. The result which appears to be least conservative is described in following manner.

The system described by equation 4.28 is asymptotically stable if;-

$$\|HC_n\|_s \, \|B_nF\|_s < \frac{\lambda_m(Q_1)\lambda_m(Q_2)}{4\lambda_M(P_1)\lambda_M(P_2)}$$

where $\|X\|_s$ is the spectral norm of matrix X, (which is equal to the square root of the maximum eigenvalue of the matrix X^TX), $\lambda_m(X)$ denotes the minimum eigenvalue of the real matrix X, $\lambda_M(X)$ denotes the maximum eigenvalue of the real matrix X, and Q_1,Q_2,P_1,P_2 are given by;-

$$\begin{aligned} A_1^TP_1 + P_1A_1 + Q_1 &= 0 \\ A_2^TP_2 + P_2A_2 + Q_2 &= 0 \end{aligned}$$

where A_1 is the dynamics matrix of the low order model and estimator, and A_2 is the dynamics matrix of the neglected part of the system.

Thus;-

$$\begin{aligned} A_1 &= \begin{bmatrix} A_r & B_rF \\ HC_r & D+GF \end{bmatrix} \\ A_2 &= A_n \end{aligned}$$

No guidance has been given for the choice of matrices Q_1 and Q_2, but the numerical example reported by Joshi and Groom [68] used unit matrices for both Q_1 and Q_2.

Again, it should be noted that this result is only a sufficient condition for stability, and to the author's knowledge, no results have yet been obtained for non-trivial necessary conditions.

4.5 Summary

The effect of using output feedback on flexible structures has been examined, and the problems of "spillover" (unpredicted and often undesirable changes in the locations of the poles of the closed loop system) have been identified.

The case when state feedback is used where the states are available with complete accuracy was examined, and it was found that closed loop stability could be guaranteed. However, this result is not particularly useful as, in practice, the states are not even measurable, so have to be estimated in some manner.

A variety of state estimator schemes were examined, the first being static estimators. These are effectively a transformation of the output information, which is in output space, to state information, which is in state space, via a fixed transformation. An advantage of this method is that it allows the results stemming from the study of output feedback to be employed, but it would be expected that this approach would be prone to parameter sensitivity. (This will be examined in a practical manner in Chapter 8.)

The three examples of dynamic state estimators subsequently examined are all well-known types of state estimator. The discussion of the Kalman Filter led to some general remarks about the choice of gains for dynamic state estimators, and suitable methods for designing high order estimators, and these remarks are examined more closely in Chapter 5.

Alternative schemes for state estimation proposed in the literature have been discussed, but not examined in detail, due to the difficulties associated with these schemes which appear to make them impractical for all but a small number of special cases.

Various works have been identified in which sufficient conditions are given under which closed loop stability of flexible structures subject to output feedback can be guaranteed. The essential aspect of these results is that a positive-definite feedback gain matrix is a sufficient condition to ensure closed loop stability.

A sufficient condition has been given to guarantee closed loop stability of a flexible structure subject to a state feedback law obtained via a dynamic state estimator, but this result is not particularly convenient to use.

To the author's knowledge, there are presently no non-trivial *necessary* conditions to guarantee stability of a flexible structure subject to feedback. Currently, the most convenient method of checking closed loop stability is to compute the closed loop poles of the system, using the highest order model available, and to examine the real parts of the poles. Obviously, any poles

with positive real parts indicate that the closed loop system is unstable.

Also, there are no guidelines at present for modifying a feedback design if the resulting closed loop system is unstable. (Except perhaps for the technique of "Innovations Feedthrough" by Balas [69], (see Chapter 5) but this does require additional sensor information to achieve stabilisation.)

Chapter 5

Feedback Law Design Techniques

5.1 Introduction

In this chapter the problems of designing feedback laws for flexible space structures are examined. Some of the methods considered are well-known methods for designing feedback laws for linear multivariable systems, others are essentially modifications to well-known methods which attempt to overcome the "spillover" problems highlighted in the previous chapter, and one method has been reported specifically for control of flexible structures.

In addition to discussing the design method itself, some of the aspects of implementing the design algorithms on a digital computer are also discussed, as the application of these algorithms to a flexible structure control problem will generally require computations that greatly exceed the practical limit of hand computation. Also the problems of "spillover" caused by the use of reduced order models in the design of the feedback laws may dictate an iterative approach to the design problem, where the reduced order model may need to be amended and the design process repeated until a useful solution has been obtained. Thus implementation of the design algorithms on a digital computer is highly desirable, if not essential.

5.2 Linear Quadratic Optimal Control

The general method of optimal control is a well-known method (see for example, [70],[71]) for designing feedback laws for multivariable systems where

the feedback law is obtained as a solution to the problem of minimising some cost function, where the cost function is chosen to define the required characteristics of the closed loop system. The optimal control problems considered here are limited to the linear quadratic, infinite time problem, that is the cost function is quadratic, and only the steady-state (or infinite time) solution is sought. The solution to this problem is known to be a linear time-invariant function of the system states (see [70], or [54]), that is state feedback via a constant gain matrix.

5.2.1 Mathematical Description

The problem described above can be stated mathematically as follows. For the system given by;-

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned}$$

a feedback law is sought which minimises the cost function;-

$$\mathcal{V} = \frac{1}{2}\int_0^\infty \left(x^T Q x + u^T R u\right) dt$$

or the essentially similar cost fuction;-

$$\mathcal{V} = \frac{1}{2}\int_0^\infty \left(y^T Q' y + u^T R u\right) dt$$

where;-

$$Q = C^T Q' C$$

It is well-known that the solution to this problem is given by;-

$$u(t) = -Fx(t)$$

where

$$F = R^{-1}B^T P$$

and the matrix P is given by the solution of;-

$$A^T P + PA - PBR^{-1}B^T P + Q = 0 \tag{5.1}$$

The above equation is generally known as the "algebraic Riccati equation".

The weighting matrices Q and R are normally chosen on an iterative basis so that the resulting closed loop system has the desired characteristics. Some aspects of the effects of the relative magnitudes of the Q and R matrices on the closed loop system are now discussed.

5.2.2 Effects of the Relative Magnitudes of the Weighting Matrices on the Closed Loop System

The effects of the weighting matrices Q and R on the locations of the poles of the resulting closed loop system are now discussed in order to understand the effects of certain techniques introduced later.

A convenient method for examining the effects of the choice of the matrices Q and R is to define them as a product of a scalar and a nominal matrix, such that;-

$$\begin{aligned} Q &= q\tilde{Q} \\ R &= r\tilde{R} \end{aligned}$$

and then to examine the effects of varying the scalars q and r.

In particular, it can be shown that as $q/r \rightarrow 0$ the optimal closed loop poles approach the locations of the stable open loop poles and the locations of the unstable open loop poles reflected about the imaginary axis. The proof of this statement is straightforward, and can be found in many control theory texts, see for example [54, chapter 8].

Note that $q/r \rightarrow 0$ implies that the control inputs are weighted much more heavily than the states. Note also that the open loop poles of a flexible spacecraft are just to the left of the imaginary axis (a very small stability margin) if damping is not ignored or at the origin of the complex plane, and thus if the control inputs are heavily weighted compared to the states, then it can be expected that the locations of the closed loop poles will not differ greatly from their open loop locations, and thus the increase in stability margin will be small. Conversely, if a large increase in stability margin is desired, then heavy weighting of the control inputs should be avoided. (See also section 5.2.4.)

At the opposite extreme, as $q/r \rightarrow \infty$ then ρ of the closed loop poles approach finite locations in the complex plane, and the remaining $n - \rho$ closed loop poles asymptotically approach infinity in Butterworth configurations of various orders and radii. The proof of this statement is less than straightforward, but it can be found in [72].

It should be noted that $q/r \rightarrow \infty$ implies that control effort is inexpensive, and thus generally results in high feedback gains and large control efforts. As spacecraft generally have finite total control effort resources (as defined by the size of their fuel tanks), control effort cannot be considered as inexpensive, and thus the control inputs should not be lightly weighted in comparision to the states. Also the high feedback gains produced by such

weighting has the tendency to make the resulting closed loop system more sensitive to the effects of noise.

Hence it can be seen from the above discussion that the choice of weighting matrices essentially has to be a compromise between the desired increase in the stability margin against the available control effort.

5.2.3 Model Error Sensitivity Suppression

This technique was postulated by Likens and Sesak [73] and is essentially a method which attempts to modify the choice of the control weighting matrix R in order to minimise the effects of unmodelled dynamics. The technique is summarised as follows.

For a system given by the equations;-

$$\begin{pmatrix} \dot{x}_r(t) \\ \dot{x}_n(t) \end{pmatrix} = \begin{bmatrix} A_r & 0 \\ 0 & A_n \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \end{pmatrix} + \begin{bmatrix} B_r \\ B_n \end{bmatrix} u(t)$$

$$y(t) = \begin{bmatrix} C_r & C_n \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \end{pmatrix}$$

for which a feedback law of the form;-

$$u(t) = -Fx(t)$$

is to be designed, which minimises the cost function $\mathcal{V}$, where;-

$$\mathcal{V} = \int_0^\infty \left(x_r^T Q_1 x_r + x_n^T Q_2 x_n + u^T R u \right) dt \tag{5.2}$$

then consider that the design is based on a low order model of the form;-

$$\begin{aligned} \dot{x}_r(t) &= A_r x_r(t) + B_r u(t) \\ y(t) &= C_r x_r(t) \end{aligned}$$

which results in a feedback law of the form;-

$$u(t) = -Fx_r(t)$$

The unmodelled dynamics are assumed to be negligible, that is $\dot{x}_n = 0$, and the cost function is modified to take account of this assumption in the following manner.

The neglected dynamics can thus be described by the equation;-

$$0 = A_n x_n(t) + B_n u(t)$$

hence;-

$$x_n(t) = -A_n^{-1} B_n u(t) \tag{5.3}$$

assuming that A_n^{-1} exists. Substituting equation 5.3 into the cost function equation 5.2, gives the revised cost function;-

$$\begin{aligned} \mathcal{V} &= \int_0^\infty (x_r^T Q_1 x_r + u^T B_n^T (A_n^{-1})^T Q_2 A_n^{-1} B_n u + u^T R u) dt \\ &= \int_0^\infty (x_r^T Q_1 x_r + u^T R' u) dt \end{aligned}$$

where;-

$$R' = B_n^T (A_n^{-1})^T Q_2 A_n^{-1} B_n + R$$

Hence this technique attempts to take account of unmodelled dynamics by effectively increasing the weighting on the control inputs in relation to the chosen weighting on the unmodelled states x_n, that is Q_2. However, it can be seen that as the weighting of the unmodelled states is increased, then there is a corresponding increase in the weighting of the control inputs. From the discussion in the previous section, it is obvious that this will result in lower performance of the closed loop system. Hence application of this method necessitates a trade-off between "spillover" reduction and performance.

An extension to this technique has been proposed by Sesak and Coradetti [39] for the design of "decentralised" controllers. As an alternative to partitioning the system as "modelled" and "neglected" parts, the partition can be arranged as a set of controlled subsystems, where a controller is designed for each subsystem. The interaction between the subsystems is assumed to be negligible by application of the above method to each subsystem in turn. This approach has also been postulated for the dual problem of designing decentralised state estimators.

However, with this approach there is no guarantee that the individually designed stable subsystems will produce a globally stable closed loop system when combined. Indeed, the manner in which global stability is obtained (see [39]) is to weight the unmodelled states extremely heavily, but from the discussion above and in the previous section, it can be seen that this results in heavy weighting of the control inputs, thus giving rise to low feedback gains and little movement of the closed loop poles from their open

loop locations. Thus if global stability is not achieved with a particular set of subsystem controllers, then the only means of correcting this problem is to re-design the subsystem controllers with heavier weighting on the unmodelled states (and correspondingly heavier weighting on the control inputs), thus producing lower feedback gains and lower performance of the subsystems. This inability to guarantee global stability severely limits the usefulness of this approach.

It should be noted that this latter technique has been referred to in the literature as "equilibrium enforcing optimal control", and "forced singular perturbation control".

5.2.4 Prescribed Stability Margin

It is possible to produce solutions to the linear quadratic design problem such that all the closed loop poles lie to the left of the line $s = -\alpha$ in the s plane. (See figure 5.1.) This is of particular interest for the control of flexible structures as the complex poles of each elastic mode lie very close to the imaginary axis, that is the stability margin is very small. Thus the ability to increase the stability margin of all the poles of the controlled part of the system may be very useful. The following analysis shows how this facility may be added to a standard linear quadratic regulator design algorithm without directly altering the solution process.

Consider the system defined by;-

$$\dot{x}(t) = (A - \alpha I)x(t) + Bu(t) \tag{5.4}$$

It is known that there exists a non-singular transformation matrix T (if $(A - \alpha I)$ has distinct eigenvalues) such that;-

$$\overline{x}(t) = Tx(t)$$

for which;-

$$T(A - \alpha I)T^{-1} = TAT^{-1} - \alpha I = \Lambda$$

where;-

$$\Lambda = diag\,\{\lambda_i\}$$

hence;-

$$TAT^{-1} = \Lambda + \alpha I = diag\,\{\lambda_i + \alpha\}$$

The solution of equation 5.4 is given by;-

$$x(t) = e^{(A-\alpha I)t}x(0) + \int_0^t e^{(A-\alpha I)(t-\tau)}Bu(\tau)d\tau$$

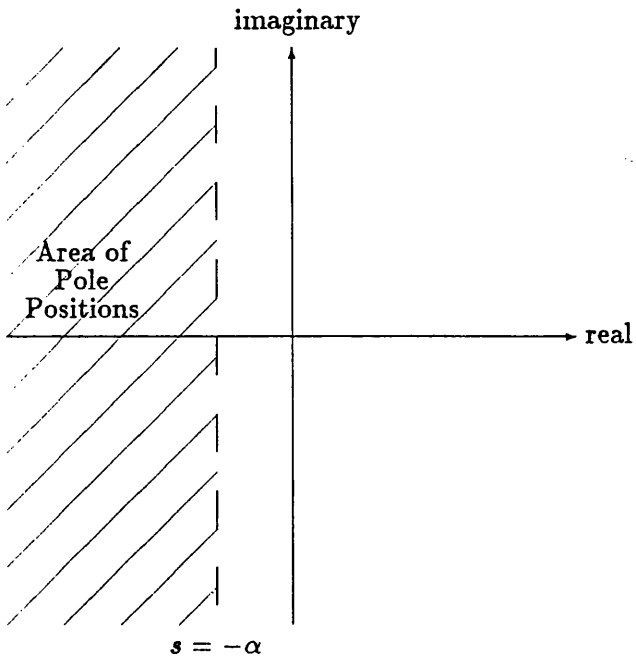


Figure 5.1: Representation of stability margin in the s plane

In the transformed basis, the solution is described as;-

$$\begin{aligned}\overline{x}(t) &= e^{diag(\lambda_i+\alpha)t - diag(\alpha)t}\overline{x}_0 + \int_0^t diag\left\{e^{(\lambda_i+\alpha)(t-\tau)}e^{(-\alpha)(t-\tau)}\right\}TBu(\tau)d\tau \\ &= e^{-\alpha It}e^{diag(\lambda_i+\alpha)t}\overline{x}_0 + e^{-\alpha It}\int_0^t e^{diag(\lambda_i+\alpha)(t-\tau)}TB\tilde{u}(\tau)d\tau \qquad (5.5)\end{aligned}$$

where;-

$$\tilde{u}(\tau) = e^{\alpha I\tau}u(\tau)$$

Returning to the original basis;-

$$x(t) = e^{-\alpha It}\left[e^{At}x(0) + \int_0^t e^{A(t-\tau)}B\tilde{u}(\tau)d\tau\right]$$

hence;-

$$e^{\alpha It}x(t) = e^{At}x(0) + \int_0^t e^{A(t-\tau)}B\tilde{u}(\tau)d\tau \qquad (5.6)$$

Recognising that equation 5.6 is the solution to;-

$$\dot{\tilde{x}}(t) = A\tilde{x}(t) + B\tilde{u}(t) \tag{5.7}$$

where;-

$$\tilde{x}(t) = e^{\alpha It}x(t)$$

and

$$\tilde{u}(t) = e^{\alpha It}u(t)$$

as in equation 5.5 above.

Substituting this change of variables into the usual linear quadratic cost function, that is;-

$$J = \int_0^\infty \left(x^T Q x + u^T R u\right) dt \tag{5.8}$$

gives the revised cost function;-

$$J = \int_0^\infty e^{2\alpha t} \left(\tilde{x}^T Q \tilde{x} + \tilde{u}^T R \tilde{u}\right) dt \tag{5.9}$$

Hence, minimising the cost function defined by equation 5.8 subject to the system defined by equation 5.4 will produce the same solution as minimising the cost function defined by equation 5.9 subject to the system defined by equation 5.7.

5.2.5 Computational Aspects

The principal computational difficulty in obtaining the solution to linear quadratic infinite time optimal control problems is the solution of the algebraic Riccati equation. Much attention has been directed at this problem and many methods have been proposed for the solution of this equation (see for example [74],[75], [76],[77]), but in most comparitive surveys the eigenanalysis method generally comes out better (see for example [78],[79],[80]), particularly for high order systems.

The principal disadvantage of most iterative methods is the requirement that the matrix A must be non-singular. (This is relaxed on certain algorithms.) This means that a solution cannot be obtained for rigid body (or rigid mode) controllers using this method. The zero eigenvalues can, of course, be approximated by small non-zero eigenvalues, but this generally results in poor convergence of the algorithm.

The eigenvalue-eigenvector method is based on the eigenanalysis of the Hamiltonian matrix (see Appendix B.3), where the solution of the algebraic

Riccati equation is formed from a partition of the eigenvector matrix. Reliable routines exist for eigenanalysis of real matrices (see Appendix B.1), even for high order problems, and thus the use of such routines enables this method to be applied to most problems to produce a solution of good accuracy.

For some problems, the computational cost (in terms of processor time) may be much greater for the eigenanalysis method than for iterative methods, but the greater reliability of the eigenanalysis method for high order problems, combined with its ability to handle rigid modes, makes it immensely suitable for application to large flexible spacecraft problems.

5.3 Independent Modal Space Control

This technique was originally postulated by Meirovitch [11] and has been applied to the control of an experimental "free-free" beam [4]. The essence of the technique is that the nth order dynamic equations, which are coupled via the input and output matrices, are decoupled by means of input and output decoupling matrices (which are defined later) into a set of $n/2$ second order systems. Obviously this reduces the design problem from one nth order problem to a set of $n/2$ separate design problems, the latter being quite trivial in comparision. This feature is very attractive but the drawback of this technique is the computation of the input and output decoupling matrices.

Consider initially the problem of decoupling the input matrix B. The problem can be stated mathematically as finding the matrix D such that the product $BD = I$, where I is the unit matrix of dimension equal to the order of the system. The necessary condition for a unique solution for matrix D is that the inverse of matrix B should exist. In fact, the input matrix for flexible structure models can be partitioned such that;-

$$B = \begin{bmatrix} 0 \\ \Phi^T \end{bmatrix}$$

where Φ is the appropriate modeshape matrix, and the corresponding partition of matrix D is;-

$$D = \begin{bmatrix} 0 & D' \end{bmatrix}$$

Thus in order to obtain a unique solution for the block D', then Φ^T must be of full rank, and with dimension that equals the number of modes to be

controlled. The practical implications of this statement is that to obtain a unique solution for the input decoupling matrix it is necessary that the number of independent actuators is equal to the number of modes to be controlled.

Clearly, a similar development can be used when considering the output decoupling matrix, which implies that to obtain a unique solution for the output decoupling matrix it is necessary that the number of independent sensors is equal to the number of modes to be controlled.

Flexible structures will generally require a large number of modes to be controlled, and the large number of control devices needed to implement this technique may prove to be prohibitively expensive. In addition to the financial limitations, the use of a large number of control devices may not be possible simply because the structure may not be able to accomodate them for physical reasons.

A variation on this technique has been suggested [81] where the need to obtain a unique solution to the decoupling matrices is relaxed, and an approximate solution is sought by the use of a pseudo-inverse. (Such a matrix inverse is sometimes referred to as a "generalised inverse" or a "Moore-Penrose inverse".)

The most reliable technique for computing a pseudo-inverse matrix is via the singular value decomposition of the original matrix. (See Appendix B.2.) However, even the use of this technique can result in significant cross-coupling terms and non-unity diagonal elements in the effective input matrix given by the matrix product BD, and likewise for the effective output matrix.

Thus the pole locations and consequently the resulting performance of the closed loop system will not be exactly as designed, the degree of unpredicted pole shifts being dependent on the "closeness" of the effective input and output matrices to unity.

Obviously the output decoupling matrix is essentially a static state estimator, as discussed in the previous chapter, and thus an alternative to decoupling the outputs with a static matrix might be to employ a dynamic state estimator as also described in the previous chapter. However, the problem of decoupling the input matrix still remains for systems with more modes requiring control than the number of available actuators.

5.4 Modern Modal Control

The technique known as "modern modal control" was originally discussed by Balas [56],[60]. The basis of modern modal control is to feed back the modal co-ordinates via a fixed gain matrix. As the modal co-ordinates are purely mathematical entities not physical variables, they have to be estimated from available physical measurements. Balas suggested the use of a dynamic estimator with the same structure as the full order dynamic estimator discussed in Chapter 4. However, on closer examination it is obvious that when the dynamic equations are written in state space form (see Chapter 2), the modal co-ordinates are actually the states of the system, so this technique is simply a state feedback technique where the states are obtained via the well-known full order asymptotic state estimator. Indeed, Balas [56] actually suggests that any of the usual state feedback design techniques such as optimal control or pole placement can be used to design a controller based on modern modal control.

An alternative interpretation of modern modal control is as a special case of the more general method of "modal control" [82],[83], where the modes are considered as second order pairs rather than individual first order modes.

The problem of "spillover" with a state feedback law and dynamic state estimator have already been discussed in Chapter 4, but Balas also postulated a variation to modern modal control [69] which attempts to overcome "spillover" problems of an initial design. The variation is essentially an additional term in the estimator equation which attempts to take account of the effects of the unmodelled dynamics on the estimator, and also a modified feedback law which is used to stabilise poles which have been destabilised by "spillover" from the original design. This modification is referred to as "innovations feedthrough", and the exact nature of this is now described.

Recall the partitioned state space equations;-

$$\begin{pmatrix} \dot{x}_r(t) \\ \dot{x}_n(t) \end{pmatrix} = \begin{bmatrix} A_r & 0 \\ 0 & A_n \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \end{pmatrix} + \begin{bmatrix} B_r \\ B_n \end{bmatrix} u(t)$$

$$y(t) = \begin{bmatrix} C_r & C_n \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \end{pmatrix}$$

and consider a feedback law designed using a reduced order model comprised of the retained modes which is of the form;-

$$u(t) = -F\hat{x}_r(t)$$

where $\hat{x}_r(t)$ is an estimate of $x_r(t)$ obtained via a state estimator of the form;-

$$\dot{\hat{x}}_r(t) = A_r\hat{x}_r(t) + B_r u(t) + H\left(y(t) - C_r\hat{x}_r(t)\right)$$

As has already been shown (Chapter 4), the closed loop poles of the full system may not be in locations as designed, and some poles may even be unstable. Balas presented conditions [69] under which the unstable poles (or poles which were significantly removed from their designed locations) could be relocated by modifying the feedback law to the form;-

$$u(t) = -F\hat{x}_r(t) - K\left(y(t) - C_r\hat{x}_r(t)\right)$$

where the matrix K is known as the "feedthrough" gain matrix, and by modifying the estimator equation by the addition of a term $TB_n u(t)$ such as;-

$$\dot{\hat{x}}_r(t) = A_r\hat{x}_r(t) + B_r u(t) + H\left(y(t) - C_r\hat{x}_r(t)\right) + TB_n u(t)$$

However, the conditons developed by Balas [69] indicate that the maximum number of modes that can be repositioned by these additional terms is equal to the number of sensors, a number that will generably be quite small. Hence this technique has serious limitations regarding its practical usage.

5.5 Robust Multivariable Servomechanism Control

The basic aim of this technique is to produce a controller which provides asymptotic regulation of the system outputs in the presence of certain perturbations in the model and in the presence of a variety of disturbances. A particularly useful facet of the underlying analysis in this method is the definition of necessary and sufficient conditions for the existance of a solution to the problem. These conditions also provide some insight into the general problem of designing controllers for multivariable systems. A rigorous analysis will not be presented here as it can be found in several sources (see for example [54, pages 299-329], or [45]), hence it is only outlined here in order to indicate the salient points. In particular, consider a linear time-invariant system described by the equations;-

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) + Ew(t) \\ y(t) &= Cx(t) + Du(t) + Fw(t) \\ e(t) &= y(t) - y_r(t) \end{aligned}$$

where $x(t) \in \mathcal{R}^n$ is the state vector,
$u(t) \in \mathcal{R}^m$ is the input vector,
$y(t) \in \mathcal{R}^p$ is the output vector to be regulated,
$w(t) \in \mathcal{R}^l$ is the disturbance vector
(which may or may not be measurable),
$y_r(t) \in \mathcal{R}^p$ is the reference signal vector,
and $e(t) \in \mathcal{R}^r$ is the vector denoting the error between the
output vector and the reference vector.

It is assumed that the disturbance vector $w(t)$ satisfies the following equations;-

$$\begin{aligned} w(t) &= C_w z_w(t) \\ \dot{z}_w(t) &= A_w z_w(t) \end{aligned}$$

where $z_w(t) \in \mathcal{R}^{n_w}$, the pair (A_w, C_w) is observable, and the initial state $z_w(t_0)$ may or may not be known. Also the reference signal $y_r(t)$ satisfies;-

$$\begin{aligned} y_r(t) &= C_r z_r(t) \\ \dot{z}_r(t) &= A_r z_r(t) \end{aligned}$$

where $z_r(t) \in \mathcal{R}^{n_r}$, the pair (A_r, C_r) is observable, and the initial state $z_r(t_0)$ is known. It is also assumed that the eigenvalues of A_w and A_r are in the closed right-half complex plane.

Now define $d_w(s)$ and $d_r(s)$ to be the minimal polynomials of A_w and A_r respectively, and let $d(s)$ be the monic least common multiple of $d_w(s)$ and $d_r(s)$. Also define $\lambda_i, i = 1, 2, \ldots q$ to be the zeros of $d(s)$, and the matrix Ω to be of companion form where the elements which are neither zero or unity are the negated coefficients of $d(s)$. (See [45].)

It can be shown [45] that a solution to the robust multivariable servomechanism problem exists if, and only if, the following conditions are satisfied;-

1. (A, B) is a stabilisable pair,
2. (A, C) is a detectable pair,
3. the number of inputs is greater than, or equal to, the number of outputs $(m \geq p)$,
4. $$rank \begin{bmatrix} A - \lambda_i & B \\ C & D \end{bmatrix} = n + p \quad i = 1, 2, \ldots q.$$

Note that conditions 1 and 2 imply that any unstable modes of the system (A, B, C, D) should be controllable and observable. Also, conditions 3 and 4 imply that none of the transmission zeros of the system (A, B, C, D) should be coincident with any of the zeros of $d(s)$. In the particular case of step disturbances and step reference inputs, the zeros of $d(s)$ will all be zero, thus condition 4 becomes;-

$$rank \begin{bmatrix} A & B \\ C & D \end{bmatrix} = n + p$$

and thus, in this case, there should be no transmission zeros of the system (A, B, C, D) at the origin. If the above conditions are satisfied, then a robust controller can be obtained. The controller actually consists of two distinct parts, a "servo compensator", which is completely determined by the nature of the disturbances and the reference signals, and a "stabilising compensator", which stabilises the overall system. A block diagram of the complete system is shown in figure 5.2.

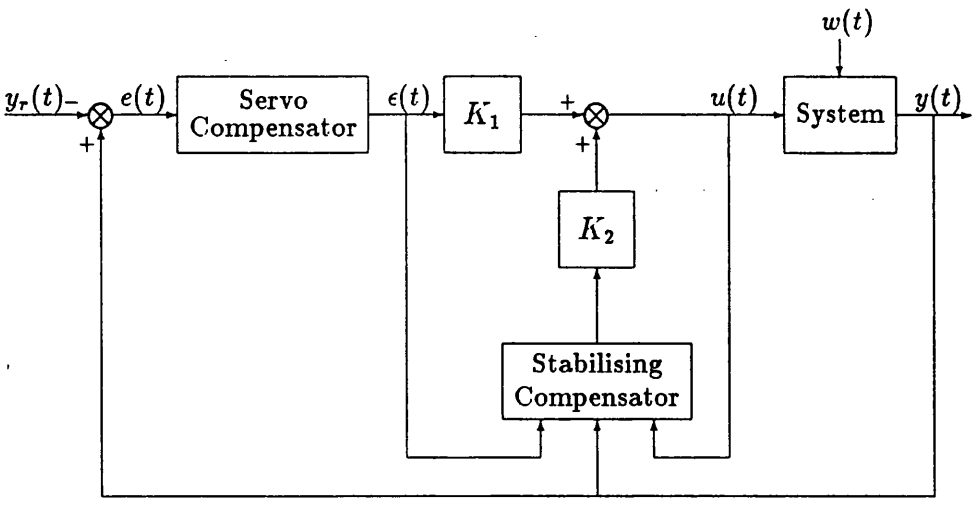


Figure 5.2: Block diagram of Robust Servomechanism Controller

The servo compensator is defined by the equation;-

$$\dot{\epsilon} = \Omega'\epsilon(t) + \Theta'e(t)$$

where

$$\Omega' = \tau \text{ block diag}\,(\Omega, \Omega, \ldots \Omega)\,\tau^{-1}$$

and $\Theta' = \tau\Theta$, and τ is a non-singular matrix, and Θ is chosen such that (Ω', Θ') is a controllable pair.

The stabilising compensator can either be of a state feedback/state estimator form, or, under appropriate stabilisability and detectability conditions [45], dynamic output feedback may be used, in a manner which stabilises the overall system. This compensator can be designed using any of the well-known multivariable control system design methods, for example optimal control, or pole placement.

It should be noted that the servo compensator is a feedback controller consisting of p unstable compensators, the dynamics of which are identical and are given by the matrix Ω. Recall that Ω represents the dynamics of the disturbances w and the reference signals y_r acting on the system. Thus the servo compensator effectively contains a model of the "external enviroment" against which it is trying to regulate. This is known as the "internal model principle" [84].

It is also worth noting that a simple structure can be obtained for the servo compensator by setting $\tau = I_{pq}$ and letting;-

$$\Theta = \text{block diag}\,(\phi, \phi, \ldots \phi) \quad \text{(p blocks)}$$

where

$$\phi = (0, 0, \ldots, 0, 1)^T \quad \text{(q elements)}$$

This technique has been applied to the control of a large flexible spacecraft [44] where output feedback (based on position and rate information) was used for stabilisation and the servo compensator was based on integral action. However, the various gain matrices were obtained by the solution of a constrained parameter optimisation problem [47]. An interesting feature of this application is that the predictable stability properties of diagonal output feedback matrices with all non-zero elements being positive (see Chapter 4) were used to enable the design to be based on a low order model comprising the rigid modes only. Thus a stabilising controller was designed for a system described by a 200th order model by using a model of only 6th order.

5.6 State Estimator Design

5.6.1 Static Estimators

The computation of a static state estimator has already been discussed in section 5.3, where it was stated that a static state estimator for the case of estimation of n states from n independent outputs can be computed via the usual matrix inversion routines (generally based on the solution of simultaneous equations). However, in the case when the number of independent outputs is less than the number of states to be estimated, it is necessary to compute a pseudo-inverse matrix. The most reliable method for computing a pseudo-inverse matrix is via the singular value decomposition of a matrix, and details of this and the computation of the pseudo-inverse from the singular value decomposition can be found in Appendix B.2.

However, it should be noted that the performance of such estimators is likely to be extremely limited, and the accuracy of the estimates is likely to be very doubtful, particularly in practical cases where the elements of the output matrix of the system (the C matrix) will not be known exactly.

5.6.2 Dynamic Estimators

5.6.2.1 Minimal Order Estimators

The details of minimal order dynamic state estimators have been discussed already in the previous chapter, hence recall the general system and estimator equations as;-

$$\begin{aligned}
\dot{x}(t) &= Ax(t) + Bu(t) \\
y(t) &= Cx(t) \\
u(t) &= -F\hat{x}(t) \\
\dot{z}(t) &= Dz(t) + Gu(t) + Hy(t) \\
\hat{x}(t) &= K_1 y(t) + K_2 z(t)
\end{aligned}$$

There are basically two approaches of selecting the matrices of the minimal order observer, one way is to assume certain values for the elements of some (or parts of some) of the matrices, and the other way is to assume a certain structure for some of the matrices. (A more thorough discussion of design methods for observers can be found in [55].

Also recall the equations governing the selection of the matrices of a minimal order observer, that is;-

$$LA - DL - HC = 0 \tag{5.10}$$

and

$$G = LB$$

In particular, equation 5.10 above has a unique solution for matrix L if the eigenvalues of matrix D are chosen to be distinct and disjoint from those of matrix A [54, p.129],[55]. A common approach to this problem is to choose matrix D and then solve for matrices L and H. One method of doing this is via a non-singular transformation Q, where;-

$$CQ = \begin{bmatrix} 0_{l,n-l} & I_l \end{bmatrix} \tag{5.11}$$

and then by partitioning the product LQ as;-

$$LQ = \begin{bmatrix} I_{n-l} & \tilde{L}_{n-l,l} \end{bmatrix} \tag{5.12}$$

In this way, the requirement on the rank of matrix L is guaranteed, and the remaining part $\tilde{L}$ is determined in terms of the transformed system matrix $\tilde{A}$, where;-

$$\tilde{A} = Q^{-1}AQ = \begin{bmatrix} \tilde{A}_{11} & \tilde{A}_{12} \\ \tilde{A}_{21} & \tilde{A}_{22} \end{bmatrix} \tag{5.13}$$

where the partitioning of matrix $\tilde{A}$ is conformal with the partitioning of matrix LQ, given by equation 5.12. Substituting equations 5.11, 5.12, and 5.13 into equation 5.10 yields two relationships, namely;-

$$\tilde{A}_{11} + \tilde{L}\tilde{A}_{21} = D \tag{5.14}$$

and

$$\tilde{A}_{12} + \tilde{L}\tilde{A}_{22} - D\tilde{L} = H \tag{5.15}$$

Thus, for a chosen matrix D, equation 5.14 can be solved for matrix $\tilde{L}$, and then equation 5.15 can be used to determine matrix H. However, with this method no guidance is available as to the form of matrix D, and also the product $\tilde{L}\tilde{A}_{21}$ is generally the product of two non-square matrices, hence equation 5.14 is often solved using the determinental relationship;-

$$\left|sI - \tilde{L}\tilde{A}_{21}\right| = \left|sI - D + \tilde{A}_{11}\right|$$

the solution of which is obviously not particularly amenable for high order systems. Minimal order dynamic state estimators designed using the above described procedure are often referred to as "parameterised" observers [55].

An alternative method where a particular form for the matrix D is assumed, is based on the pole assignment algorithm. Initially the equations are transformed to the "observable canonical" form [38], [54], where the transformed system matrix is essentially block diagonal, with the blocks taking the companion form. The matrix D for the observer is now specified to have similar form and thus specification of its desired eigenvalues completely determines the matrix. From this, the matrix L in the transformed basis $\tilde{L}$ can be assigned, and then equation 5.10 in the transformed basis, that is;-

$$\tilde{L}\tilde{A} - D\tilde{L} - \tilde{H}\tilde{C} = 0$$

can be solved for $\tilde{H}$ (noting that this particular form allows a conveniently simple solution). The matrices $\tilde{L}$ and $\tilde{H}$ can then be transformed back to the original basis, giving the required matrices L and H.

In both the above methods, the matrices K_1 and K_2 can be found from;-

$$\begin{bmatrix} K_1 & K_2 \end{bmatrix} = \begin{bmatrix} C \\ L \end{bmatrix}^{-1}$$

The latter procedure outlined above appears to be more attractive than the former procedure outlined above from a computational viewpoint, but it does have one particular drawback. The transformation of the system equations to observable canonical form requires the construction of the "observability matrix" Φ_o [38] for the system, where Φ_o is given by;-

$$\Phi_o = \left[C^T, A^T C^T, (A^T)^2 C^T, \ldots (A^T)^{n-1} C^T\right]^T$$

where n is the order of the system. Obviously, the computation of this matrix for high order systems will produce numerical problems as any ill-conditioning of the matrix A will be magnified in the computation of the powers of A. This can lead to severe ill-conditioning of the resulting observability matrix, and also significant numerical inaccuracies. Even if the ill-conditioning is not too severe, the numerical inaccuracies are carried into the computation of the estimator matrices, which can lead to poor performance of the resulting estimator.

5.6.2.2 Full Order Estimators

The details of full order dynamic state estimators have already been discussed in the previous chapter, hence recall the system and estimator equations as;-

$$\begin{aligned}\dot{x}(t) &= Ax(t) + Bu(t)\\ y(t) &= Cx(t)\\ u(t) &= -F\hat{x}(t)\\ \dot{\hat{x}}(t) &= A\hat{x}(t) + Bu(t) + H\left(y(t) - C\hat{x}(t)\right)\end{aligned}$$

Generally, the dynamic characteristics of a state estimator are of most significant importance and thus design methods which include the facility to locate the poles at desirable locations are most useful in dynamic state estimator design. Hence a logical first choice of a design method is some form of pole placement algorithm. In a similar manner to that described above, the equations can be transformed to the observable canonical form, and the matrix D can also be specified in this form.

Recall from the previous chapter, that the matrix D is now of the same order as that of the matrix A, and that $L = I$, and thus the equation to be solved is now;-

$$\tilde{A} - D - \tilde{H}\tilde{C} = 0 \qquad (5.16)$$

Thus equation 5.16 can be solved for matrix $\tilde{H}$, and thus H obtained by transforming back to the original basis. In common with the procedure outlined in the previous section, the only computational problem with this procedure is the computation of the observability matrix Φ_o, which is a necessary prerequisite to the computation of the transformation matrix which will transform the system to the observable canonical form.

Another problem common to design methods based on pole placement is that it is obviously necessary to know *a priori* where the poles are to be placed. At present, the only guidelines available as to the "best" locations for poles of dynamic state estimators is that they should be faster than the system poles and reasonably well-damped so that the estimates converge quickly to the true values. This effectively defines a minimum time-constant for the estimator poles, and a minimum damping for the poles. It is also known that systems with very fast poles tend to be more sensitive to the effects of noise than systems with slower poles, so this effectively means there should be an upper limit to the time-constant for the poles. These three limitations can be interpreted as boundaries in the complex plane, as

shown in figure 5.3, which defines an area in the complex plane in which the estimator poles should lie. Another aspect of this is that it appears [18], (but it has not been categorically proven) that robustness properties (in the sense of sensitivity to model mis-match) of dynamic state estimators may be linked to the relative positions of the poles of the estimator, in that if the poles are closely grouped, then the performance of the estimator is more sensitive to model mis-match than if the poles are well-spaced. Thus it can be seen that the selection of pole locations for dynamic state estimators has to be considered carefully, and the process becomes increasingly difficult as the order of the estimator increases.

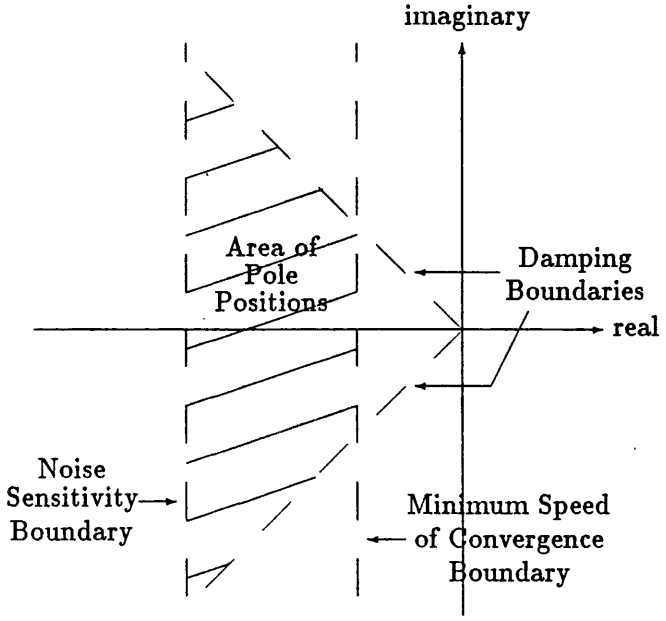


Figure 5.3: Boundaries for observer pole locations in the s plane

An alternative method for the design of full order dynamic state estimators which avoids the need to specify the pole locations *a priori* has already been suggested in the previous chapter, where the gains (matrix H) are chosen as the steady-state Kalman Filter gains. A computional procedure which facilitates the computation of the steady-state gains "off-line" is the dual of the Linear Quadratic Regulator problem discussed in section 5.2. In

particular, the gain matrix H is given by;-

$$H = PC^T R^{-1}$$

where the matrix P is given by;-

$$AP + PA^T + Q + PC^T R^{-1} CP = 0 \tag{5.17}$$

This is sometimes referred to as the Linear Quadratic Gaussian solution [71]. In the case of the Kalman Filter, the matrices Q and R depend on the characteristics of the noise sources, but in the noise-free case, they may be used simply as weighting matrices, in a similar manner to that described in section 5.2.2, in order to direct the solution of equation 5.17 to achieve certain dynamic characteristics in the resulting estimator.

The numerical solution of equation 5.17 has already been discussed in section 5.2.5, and is examined in greater detail in Appendix B.3.

Note that the resulting state estimator is essentially the same as the Wiener Filter [85] originally postulated via a Fourier analysis approach, although the original work applied only to single-input, single-output systems, and the design algorithm was very different.

5.7 Other Techniques

5.7.1 Multivariable Frequency Domain Techniques

Two well-known techniques for designing controllers for linear multivariable systems are the Nyquist Array (or the Inverse Nyquist Array) technique [86], [87], and the Characteristic Loci technique [88], [89], but both of these techniques are subject to limitations when applied to the design of controllers for flexible structures. The reason for this is that both techniques are based in the frequency domain, and as has already been noted in Chapter 2, the extremely low levels of damping give rise to problems of scaling in the plotting of frequency-dependent gain curves. However, with the development of suitable software incorporating "zooming" facilities to observe detail in plots, this problem could be overcome.

Another limitation is that the system should be square, that is the number of inputs must equal the number of outputs. For "shape" control problems, the "square" requirement is likely to severely limit the usefulness of these techniques. For attitude control problems, this is probably not a limitation, as an initial design can be carried out for the three axes of a "pointing" problem, for example, and the initial controller design can then be

adjusted by the addition of any further sensor information. For example, if the initial design assumed that only position information was available, then the controller would need to incorporate some form of derivative action, the magnitude of which could be identified by the initial design, but if subsequently rate information became available, then rate feedback could be used instead of the derivative action, where the gain of the rate feedback loop would be set to provide the same effect as the derivative action. Thus the required characteristics of the controller could be identified, although the actual implementation may differ.

A major drawback of these techniques is that for practical usage they require reasonably sophisticated graphics software for displaying and manipulating the various curves. Although the implementation of such software would be relatively straightforward, it would be very time-consuming, and for this reason, these techniques were not examined further.

5.7.2 Positivity

Another technique that has been reported in the literature for application to flexible structures is a technique known as "Positivity" [90],[91]. Again, this is a frequency domain technique, but which is based on positive real transfer function matrices. This technique can be used to obtain bounds for controller gains to ensure stability of the closed loop system in the presence of parameter variations, but the variations must be known or assumed *a priori*.

The theory is based on the positivity of operators, the following definitions being fundamental;-

1. Square transfer function matrix $G(s)$ is Positive Real if;-
   - $G(s)$ has real elements for real s
   - $G(s)$ has elements which are analytical for $Re[s] > 0$
   - $G^*(s) + G(s)$ is positive semi-definite for $Re[s] > 0$ (where $G^*(s)$ is the complex conjugate transpose of $G(s)$)
2. Square transfer function matrix $G(s)$ is Strictly Positive Real if;-
   - $G(s)$ has real elements for real s
   - $G(s)$ has elements which are analytical for $Re[s] \geq 0$
   - $G^*(j\omega) + G(j\omega)$ is positive definite for all real ω

For the system shown in figure 5.4, where $H(s)$ and $G(s)$ are square transfer function matrices, the system is asymptotically stable in the input/output sense if at least one of the matrices is strictly positive real, and the other is positive real. The proof of this is shown in [90].

The requirement of plant matrix $G(s)$ to be at least positive real is rather restricting, so a technique of "embedding" can be used to introduce new transfer functions without altering the overall stability characteristics of the system. Details of the design method together with a simple example can be found in Appendix C.

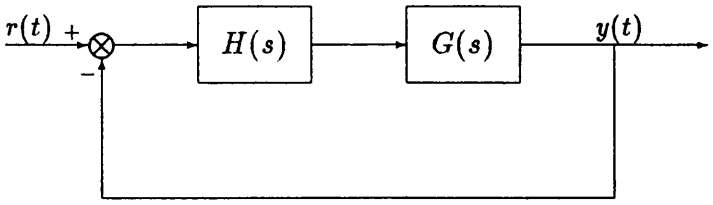


Figure 5.4: Schematic of transfer function matrices for Positivity concept

A particular drawback of this technique is that it cannot handle poles on the imaginary axis, so rigid modes have to approximated in some manner, generally as a pair of very low frequency complex poles. However, as the degree of approximation is reduced, the rigid modes dominate the design method, and so the bounds obtained for controller gains depend on the approximation of the rigid modes [92]. The simple worked example in Appendix C shows the method is rather conservative, and this has been confirmed by a more thorough study in [92], where this technique was applied to a 2-mode approximation of the Olympus satellite, using both a single axis model and a three axis model.

Thus the conservatism of this technique, and its difficulty in handling rigid modes, and to some extent the requirement that the system should be square, makes it unattractive as a design technique for flexible spacecraft.

5.8 Summary

Several techniques for designing feedback laws for flexible spacecraft have been discussed. The computational aspects have also been discussed because

the nature of the problems of designing controllers for flexible spacecraft dictates that hand computation is generally impractical, and that frequently, a design technique may need to be carried out several times in order to obtain a useful solution. Thus computer-aided design routines are, for all practical purposes, essential.

The Linear Quadratic Optimal Control technique hinges on the solution of the algebraic Riccati equation, and an algorithm for solving this equation based on an eigenanalysis of the Hamiltonian matrix (see Appendix B.3) has been found to give reliable results, even for high order systems. The effects of the relative magnitudes of the weighting matrices Q and R have also been examined, thus enabling the understanding of the effects of the technique known as "model error sensitivity suppression", which claims to reduce the effects of "spillover", although in practice it achieves this merely by reducing the control action, with subsequent reduction in overall system performance.

The technique of Independent Modal Space Control appears initially to be extremely attractive as it enables an *nth* order design problem to be considered as $n/2$ separate second order design problems. However, the requirement of having the number of sensors and the number of actuators equal to the number of modes to be controlled in order to obtain a unique solution is very restrictive, and thus greatly limits its usefulness. The relaxation of the need for a unique solution enables pseudo-inverse matrices to be used, but the accuracy of the decoupling using these matrices is doubtful.

The technique known as Modern Modal Control is merely a title which describes not so much a technique, but a *class* of controllers. This class of controllers is based on state feedback where the states are obtained using a dynamic state estimator. A technique for overcoming "spillover" problems in closed loop systems incorporating a controller belonging to the class of modern modal control has been discussed whch is referred to as "Innovations Feedthrough". However, this technique has a major limitation in that the number of modes which can be relocated is equal to the number of available sensors.

Another technique which to some extent describes a *class* of controllers is Robust Multivariable Servomechanism Control. Controllers resulting from this approach have two parts, a servocompensator to provide the regulating action, and a stabilising compensator to stabilise the overall system. The servocompensator is derived solely from information about the disturbances and reference signal dynamics, whereas the stabilising compensator can be designed using any multivariable control system design method, and is re-

quired to stabilise the overall system.

Any controller requiring information about the states of the system will dictate the use of a state estimator of some form, and various methods for designing state estimators have been discussed. The methods based on pole-placement algorithms are hindered by the need to compute the observability matrix Φ_o prior to transformation of the equations to observable canonical form, which can lead to large inaccuracies, or even algorithm failure, for high order sytems, or numerically "stiff" systems. One method which is not prone to such problems is the design of a steady-state Kalman Filter, which can be obtained via the solution of an algebraic Riccati equation. As has already been mentioned, this can be achieved reliably, even for high order systems, by the use of an routine based on the eigenanalysis of a Hamiltonian matrix.

Thus it can be seen that the choice of methods for designing controllers for flexible spacecraft is likely to be limited by the computational details of the methods rather than anything else. Indeed, it has been found (see Chapter 8) that for many problems, the majority of design methods simply fail to obtain a solution, or the degree of inaccuracy makes the solution useless.

Chapter 6

Experimental Rig – Hardware

6.1 Introduction

The purpose of developing the experimental rig was to demonstrate the application of the methods of designing control systems for flexible structures to a real structure, not just in the form of simulations, but also in the form of real-time control. Consequently, it was necessary to design and build a structure on which the demonstrations could be based, and control devices (actuators and sensors) with which to control the structure, and a computing system with which to implement the control laws. Following modern practice, the computing system is digital, and because the operating system supports high-level languages, it has also been used for software development and simulation work. In this chapter, the hardware aspects of the experimental rig are discussed, and the software which supports this hardware, and which is used to provide various other facilities is discussed in the next chapter. An overall schematic layout showing the interconnection of the hardware which comprises the experimental rig is shown in figure 6.1. The photograph shown in figure 6.2 clearly shows the major components of the rig. The development of all aspects of the experimental rig is now described.

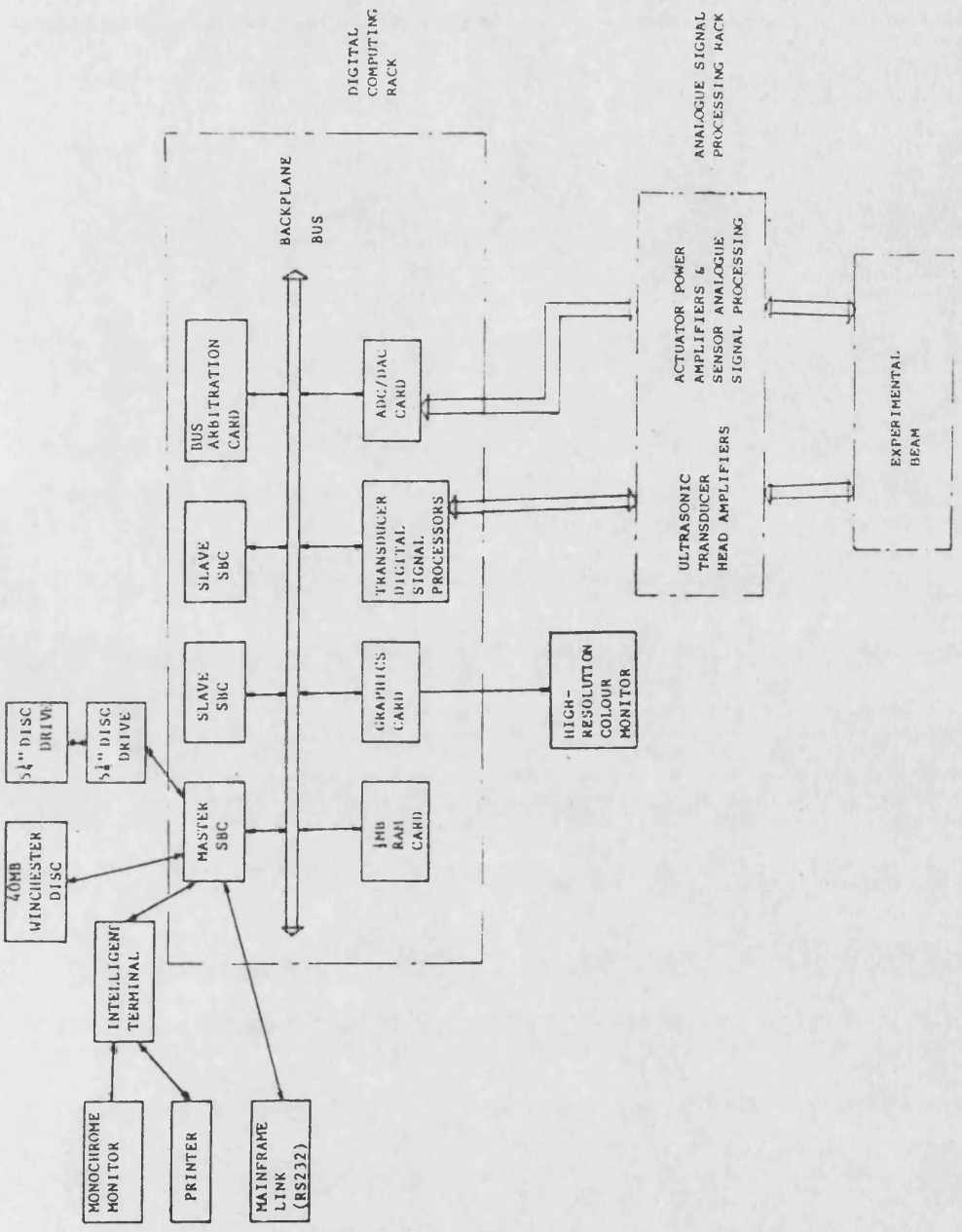


Figure 6.1: Schematic of hardware comprising the experimental rig

Figure 6.2: Photograph showing overall view of experimental rig

6.2 The Structure

It was desired that the structure should be as good a representation of a flexible spacecraft as was reasonably possible, within the various constraints applying to a laboratory rig. The effects of gravity can only be avoided by considering displacements in the horizontal plane, and the desire to include a rigid mode constrained the structure to be essentially a one-dimensional problem, to avoid the need for expensive low-friction gimbal bearings.

Initially it was necessary to be able to predict the resonant frequencies of the structure analytically, and hence a relatively simple uniform structure was dictated. These factors led to the decision to that the structure should be beam-like, supported at its midpoint by a pivot, and orientated such that displacements in one dimension of the horizontal plane only were of interest (see figure 6.3). Thus the structure has one rigid mode (rotation about the pivotal axis) plus its elastic modes.

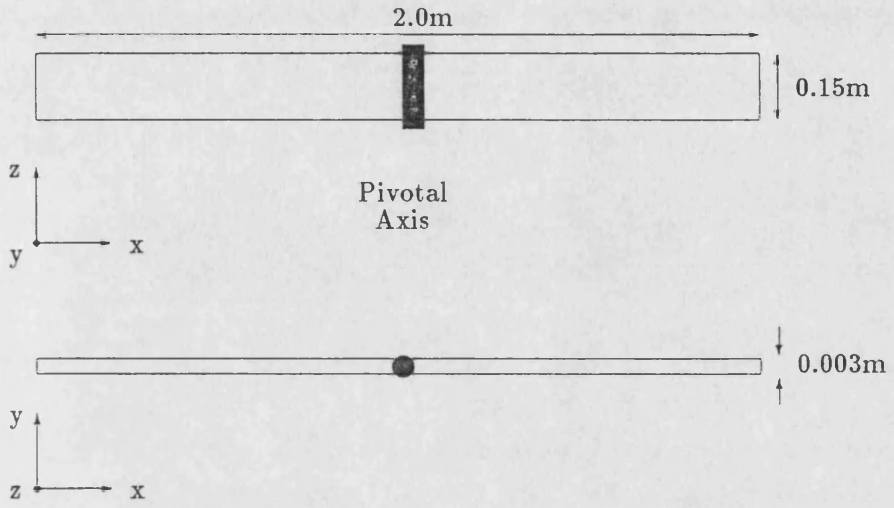


Figure 6.3: General layout of beam

Originally, it was desired that the beam should be at least 4 metres long in order that the requirements on the speed of the computing system should not be too excessive. However, because laboratory space was at a premium, and that material of this size was not readily available, this was not possible. Thus, due to its availability, the beam was constructed from two pieces of aluminium alloy, each of dimension 1.0m $\times$ 0.15m $\times$ 0.003m, which were joined together at the pivot giving an overall length of 2.0m.

The frequencies of the elastic modes of a uniform beam are known to be regularly spaced [11],[12],[93], but in general, the elastic modes of a flexible spacecraft are not. In order to alter the frequency distribution of the elastic modes, holes of 0.05m diameter were cut in the beam at intervals of 0.1m which facilitated the mounting of steel weights, the position of which could be chosen to alter the frequencies of the elastic modes and to alter their distribution. This facility to alter the properties of the structure could also be used to investigate the sensitivity of control laws to model parameter variations. The holes in the structure were also intended for the mounting of vibration control actuators, which are described in the next section, and consequently, the mass of the steel weights was chosen to be the same as the mass of the vibration actuators (about 0.8Kg) so that actuators and masses could be interchanged with minimal alteration of the characteristics of the

structure. These details of the actual beam can be seen in the photograph shown in figure 6.4.

6.3 Actuator Development

Several different types of actuators were considered, some being pursued further than others. The details of these various types of actuators considered for the experimental rig are now discussed.

6.3.1 Torque Actuator

The principle actuator required is a torque actuator to provide torques about the pivotal axis of the structure, whereas force actuators were envisaged to assist with vibration control. The development of a torque actuator was based on the principle of a reaction wheel, where a d.c. motor was used to accelerate (or decelerate) an inertia wheel which is mounted coaxially with the support pivot of the structure. An inertia wheel was constructed from mild steel in the form of a cylinder of 0.05m diameter and 0.05m length, and was coupled to the shaft of an available miniature d.c. motor. However, the power output of the d.c. motor was not sufficient to provide adequate torque levels to excite the elastic modes. Due to financial and time limitations, no further development work has been carried out on this actuator, but the obvious next step would be the acquisition of a more powerful motor, and the re-design of the inertia wheel accordingly.

6.3.2 Force Actuators

Some time was spent developing a force actuator that could be used for vibration control of the structure. The basic principle relies on the reaction force exerted by the actuator body when accelerating a mass. The acceleration is provided by an electromagnetic field created by the excitation of a coil. The first version that was constructed is composed of a steel ball bearing running in a brass tube with end stops, where the two coils (one for acceleration in each direction) are mounted on the outside of the brass tube. (See the sectional sketch in figure 6.5.) The coils comprise about 330 turns of 0.022in enamelled copper wire.

The use of a steel ball resulted in low friction levels which allows a large range of driving signals, from the saturation limit of the power amplifiers down to very small levels just sufficient to overcome friction. However the

Figure 6.4: Photograph showing experimental beam during development

average position of the ball is extremely sensitive to the mismatch between the two electromagnetic fields created by the coils. The law governing the force of attraction is an inverse square law with respect to distance [94], and thus, even when driven by a signal with zero mean, the coil with greater "efficiency" quickly "wins" the ball, causing it to oscillate against the end stop in the tube. The addition of a feedback loop to regulate the position of the ball was considered, but in the absence of any convenient signals for feedback, this idea was not pursued.

The second version constructed attempted to overcome the problem with the average position of the mass by the use of two linked masses. The moving mass in this version is configured as a dumbell, with the central shaft running in a brass bush. (See sectional sketch in figure 6.6.) This version did indeed overcome the stability problem of average position of the mass, but the friction of the central shaft in the brass bush significantly reduced the useful

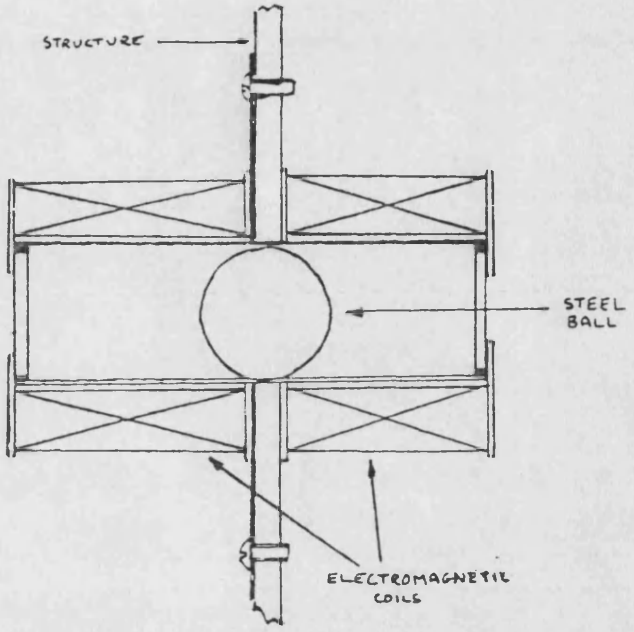


Figure 6.5: Sectional sketch of *rolling-ball* force actuator

range of output levels.

Subsequently, a third version was constructed, which attempted to combine the best features of both previous versions. Thus the third version has a moving mass in the form of a dumbell, but it is supported by miniature pony trucks. (See sectional sketch in figure 6.7.)

This version was designed to work over a frequency range of approximately 1Hz to 30Hz. The moving mass and the distance it travelled needed to be kept to a minimum in order to minimise the effects on the dynamics of the structure. It is now shown that these are conflicting requirements.

Assuming that the required actuator force is sinusoidal, and assuming zero initial conditions, then;-

$$\ddot{x}(t) = \frac{A}{M} sin(\omega t)$$

where;-

A is the peak force,
ω is the cyclic frequency of the force,

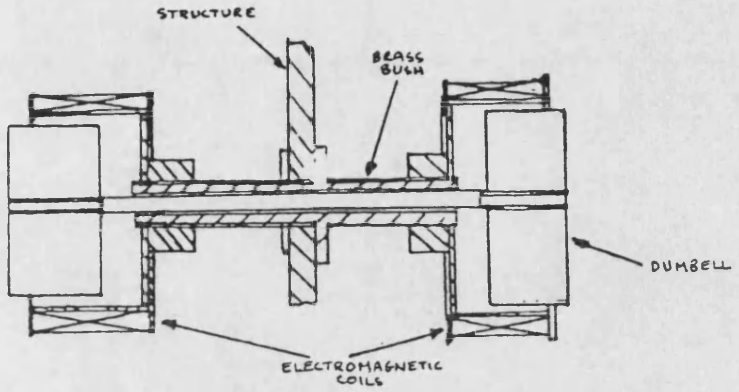


Figure 6.6: Sectional sketch of *dumbell* force actuator

M is the mass of the moving shuttle,
and $x(t)$ is the displacement of the shuttle.

Integrating with respect to time twice, gives;-

$$x(t) = -\frac{A}{M\omega^2} sin(\omega t)$$

hence the peak displacement $\hat{x}$ is given by;-

$$\hat{x} = \frac{A}{M\omega^2}$$

Thus, for a given operating frequency, if it is desired to reduce the mass, then the peak displacement will necessarily increase.

Another difficulty arises with the fact that each mass is only pulled to the centre of the appropriate coil (the position of minimum inductance [94]), and the mass at the other end of the dumbell must remain within the "reach" of the electromagnetic field created by its coil. Thus the final design resulted in a moving mass of about 0.5Kg, a stroke length of just under 0.05m, with a corresponding coil length of 0.1m. The overall length of the resulting actuator is 0.22m.

This version appeared to work quite well initially, but the reliability of the miniature pony trucks is poor, resulting in frequent bearing collapse and consequent failure of the actuator.

To assist the development of these actuators, it was necessary to design a power amplifier which behaved as a voltage-controlled current source. The circuit diagram of this amplifier is shown in figure 6.8. The amplifier is

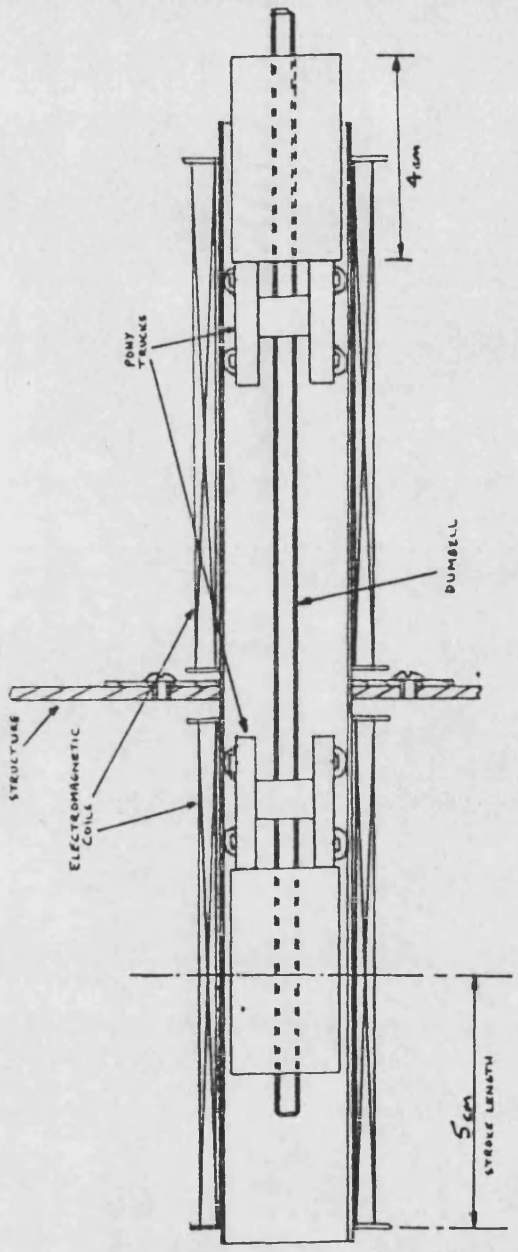


Figure 6.7: Sectional sketch of final force actuator

designed to produce a current in either coil of an actuator of up to 10 Amps, proportional to the magnitude of the voltage at its input, and the particular coil of the actuator being selected by the sign of the voltage at its input. The various trimmers are used to minimise the differences between different coils and output transistors.

The development of this actuator has led to some thought about its suitability, in a modified form, to the problem of structural damping of real flexible space structures. In particular, consider a device constructed of two parts, a magnetic core and an electromagnetic coil, where the magnetic core is connected to one point on the structure, and the electromagnetic coil is connected to another point on the structure such that when the structure is undeformed, the core lies at the point of minimum inductance of the coil (see figure 6.9). When the structure deforms, the core will move away from its nominal position, but if the coil is now excited, then regardless of the direction of the excitation current, a force will be exerted on the core in such a manner as to attempt to return it to the position of minimum inductance. Thus it is feasible that such a device may be used to effectively increase the mechanical stiffness of a structure, although no work has been done to examine this possibility.

6.3.3 Air Jet Actuators

The development of air jet actuators was also considered, inspired by the current use of gas jets on spacecraft. Initial calculations using the equation for variable density flow through a nozzle [95, page 410] suggested that a nozzle reducing from 3.0mm diameter to 0.5mm diameter fed from an air supply of about 40p.s.i.g. would produce useful force levels. However, in the absence of a suitable air supply in the laboratory, this idea could not be pursued any further.

6.4 Sensor Development

As for the actuators, a variety of different types of sensors were considered for the experimental rig, and again, some were pursued further than others. The details of the various types of sensors considered are now discussed.

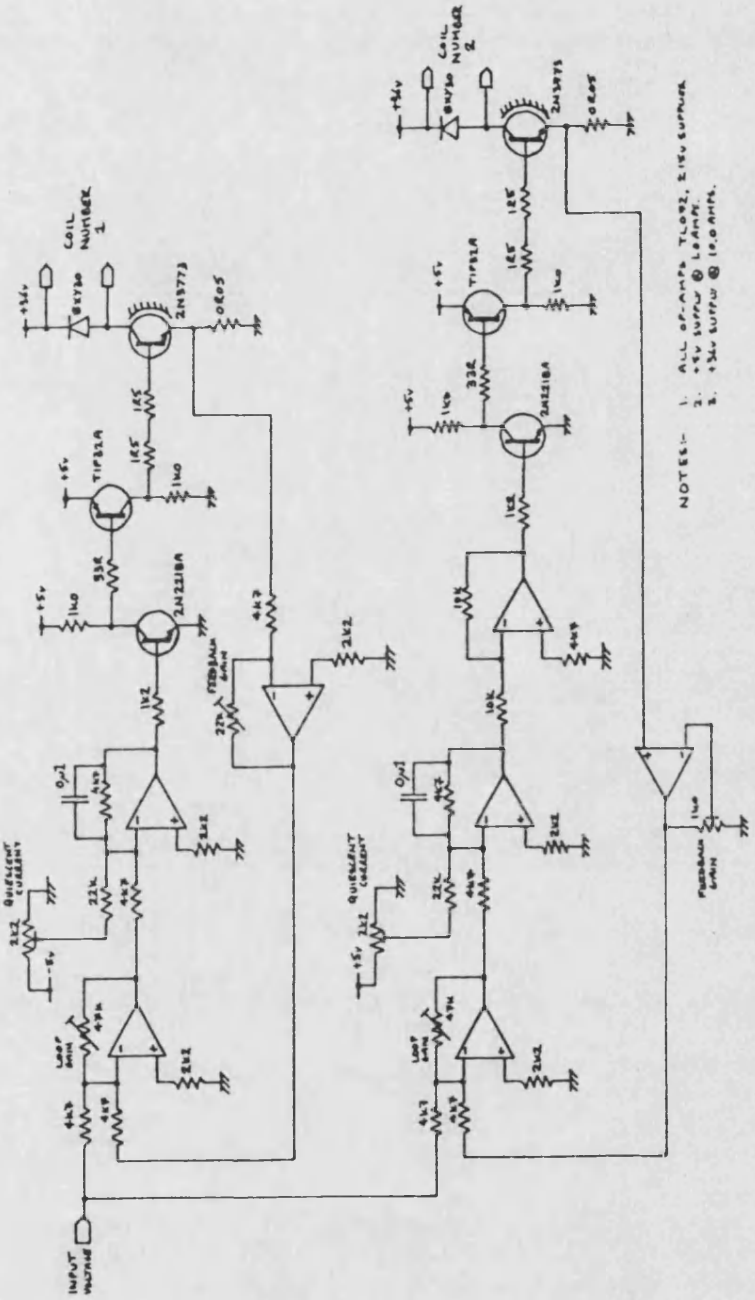


Figure 6.8: Circuit diagram of force actuator power amplifier

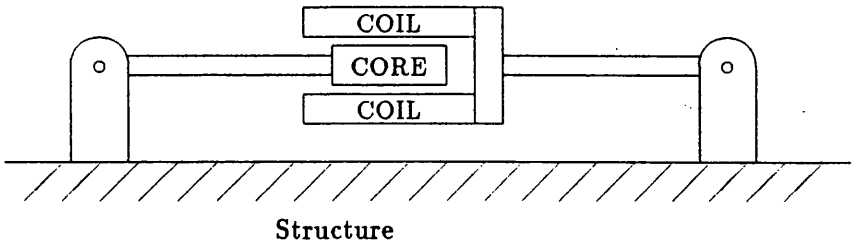


Figure 6.9: Sectional sketch of proposed electromagnetic active stiffener

6.4.1 Position Sensor

The principal sensor required is a position sensor to measure rotation of the support pivot so that the overall position of the structure can be determined. A high-resolution servo potentiometer directly coupled to the pivot shaft meets this requirement. It was anticipated that suitable signal processing could be used to extract rate information.

The two ends of the servo potentiometer are directly connected to a regulated 30 Volt d.c. supply. Thus the wiper of the servo potentiometer provides a d.c. voltage proportional to its position. This analogue voltage obviously requires conversion to a digital signal for processing by the digital computer, and this is done using a analogue to digital converter, which is described in section 6.5.2.1.

6.4.2 Displacement Sensors

In order to validate models, and to observe detailed behaviour of the structure, it was desired to be able to measure the displacements of the structure at various locations along its length, simultaneously. Two approaches were considered for this, the first being based on ultrasonic displacement transducers, and the second being based on strain gauges.

6.4.2.1 Ultrasonic Displacement Sensors

These sensors were originally developed at the University of Bath for measuring the suspension height of a magnetically-levitated vehicle, and details of this original application can be found in [96], although the original design

has been modified to suit this particular application. The circuit diagram for a single transducer is shown in figure 6.10, and with reference to this, the mode of operation will now be described.

A 10.24MHz clock signal is generated by a voltage controlled oscillator (1/2 of a 74S124) and a 10.24MHz crytal, thus providing a stable reference frequency. The 10.24MHz clock is divided by 256 using a dual 4-bit binary counter (74LS393) to produce a 40KHz signal which is buffered and sent to the analogue processing rack via line drivers, where it is again buffered and used to drive the ultrasonic transmitter. This signal is also divided by 2 (to 20KHz) using a D-type flip-flop (half of a 74LS74), and consequently fed to a state machine [97].

The ultrasound signal emitted by the transmitter is reflected off the target, and converted back to an electrical signal by the ultrasonic receiver. The signal produced by this device is a low amplitude sine wave, which is then amplified by the discrete transistor (BC109) and associated components. The discrete components have been chosen such that the transistor amplifier has a frequency response characteristic of a bandpass filter, centred around 40KHz. The amplified signal is ac-coupled to the input of a voltage comparator (TL710), which converts the sine wave signal to a TTL compatible logic signal. This signal is buffered and sent back to the digital processing circuit via line drivers, where it is divided by 2 (to 20KHz) using a D-type flip-flop (the other half of a 74LS74), and consequently fed to the state machine for comparison with the reference signal. The other two inputs to the state machine are produced by the result of the lower byte (eight bits) of the output and are used for indicating an up count or a down count for the upper byte, as will be explained later.

The state machine is constructed from an octal D-type flip-flop (74LS374) and an 256 $\times$ 4 bipolar programmable read-only memory (82S129). The lower 3 bits of the state machine are decoded by a 3-to-8 line decoder (74LS138) such that output 6 (pin 9) produces a pulse which has a width proportional to the phase difference between the reference signal and the received signal. This pulse is used to enable a dual 4-bit binary counter (74LS393 arranged as an 8-bit counter), which counts the number of pulses of the 10.24MHz clock. The trailing edge of the enabling pulse is used to latch the output of the 8-bit counter into an octal D-type flip-flop (74LS374), thus producing the lower byte of the data word. The 8-input NAND gate (74LS30) which is connected to the output of the 8-bit counter is used to detect when an overflow or an underflow occurs and provides inputs to the state machine accordingly (the two inputs referred to earlier).

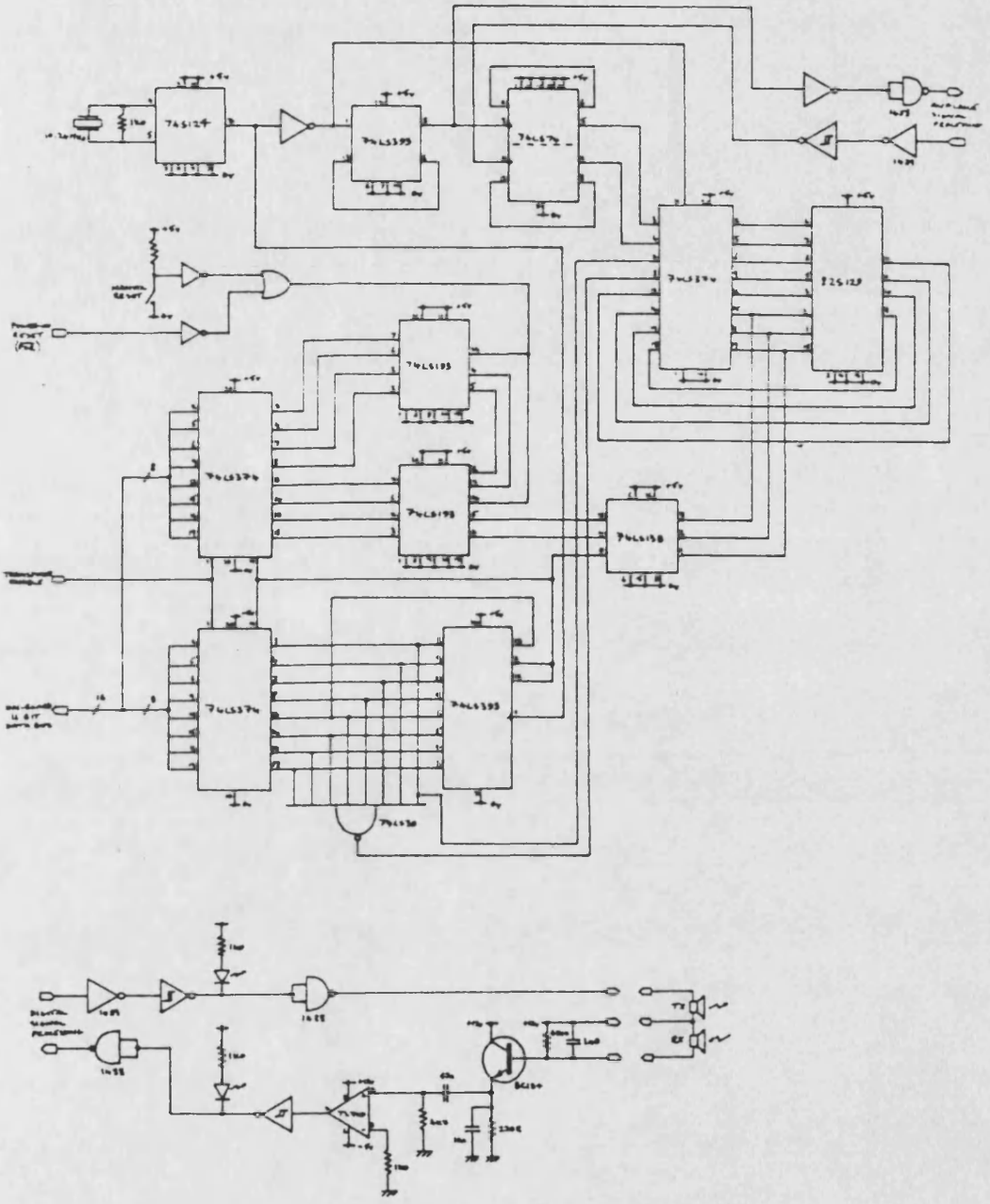


Figure 6.10: Circuit diagram of ultrasonic transducer

Returning to the outputs of the 3-to-8 line decoder, outputs 2 and 5 (pins 13 and 10) produce pulses each time an overflow or an underflow occurs. Output 2 produces a pulse each time an overflow occurs, and output 5 produces a pulse each time an underflow occurs. These pulses are counted by a pair of 4-bit up/down counters (2×74LS193) connected to form an 8-bit up/down counter, which provides an extension to the lower byte. The output of this 8-bit up/down counter is latched into an octal D-type flip-flop (74LS374), thus producing the upper byte of the data word.

When the outputs of the two octal D-type flip-flops holding the 16-bit data word are enabled by the address decoding logic (which is described in Section 6.5.2.2), the 16-bit word is made available to the backplane data bus where it can be read by the single board computer which generated the address.

It is useful to obtain a scaling factor to equate the binary word to a physical distance, and this is now considered. The velocity of sound in air is 331 metres/sec, and from the well-known formula $V = f\lambda$, the wavelength of the 40KHz signal is 8.275×10^{-3}metres. This wavelength is doubled by the division to 20KHz, and then this signal is partitioned by the 10.24MHz clock, that is the 40KHz signal is effectively partitioned into 512 parts, which corresponds to a distance of $8.275\times10^{-3}/512$metres which is 16.162μmetres (0.016162 mm). Thus the least significant bit corresponds to a distance of 16.162μmetres, and the most significant bit corresponds to 0.5296 metres, thus giving a range of just over one metre. It should be noted that this transducer can only provide a measure of the displacement relative to its starting point, and thus a manual reset is provided to set the upper byte to zero, and the software used to process the data from these sensors takes this into account. Also, this measurement is of the length of the transmission path of the signal, so further processing is required to extract the actual perpendicular distance between sensor and structure which depends on the geometry of the sensor and its displacement from the structure (see figure 6.11).

6.4.2.2 Strain Gauge Sensors

The idea of using strain gauges to measure displacement of a structure is well-known, and is based on the fact that for small displacements, strain is proportional to displacement, or to be more exact, strain is proportional to deformation (see Appendix A.1). Thus the relative position of the structure could, in theory, be reconstructed from strain information.

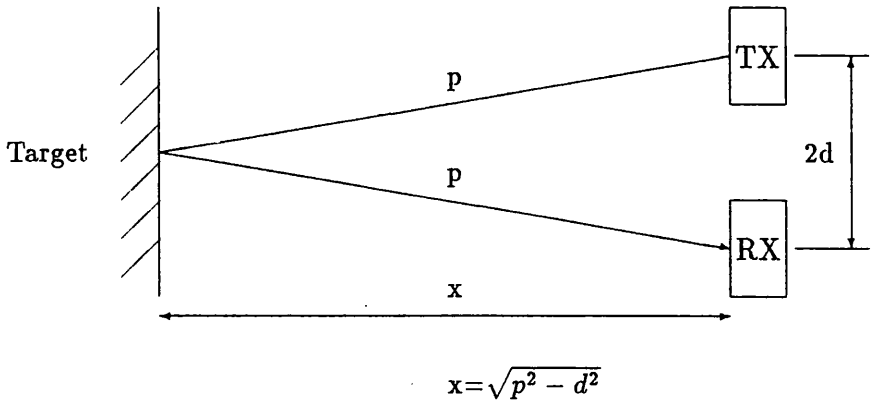


Figure 6.11: Sketch showing geometry of ultrasonic sensors

However, a strain gauge at a particular location only produces information about deformation of the structure at that point, and consequently any position calculations require some reference point to work from. If information about the elastic deformation of the structure only is required, then the deformation at any point can be measured by affixing a strain gauge at that point, and measuring the change of resistance of the strain gauge. The change in resistance is converted to a d.c. voltage by means of the usual bridge network and differential amplifier (see figure 6.12).

If it is required to measure the displacement of the structure relative to a known reference frame, for example the deformation of the structure from its nominal shape, then strain measurement is much less suitable. This is because the only fixed point on the structure which can act as a reference is the support point, which in the case of the experimental beam, is the support pivot. Thus, to obtain the displacement of the structure at any point would require the computation of the relative displacement (the deformation) of the structure at many points between the fixed point and the point under consideration.

This approach has several practical difficulties. Firstly, it would require a large number of closely spaced strain gauges plus the associated bridge circuits and amplifiers in order to minimise the error caused by unmeasured deformations between measured points. Secondly, even if sufficient gauges are available, any practical measuring device will have some finite error as-

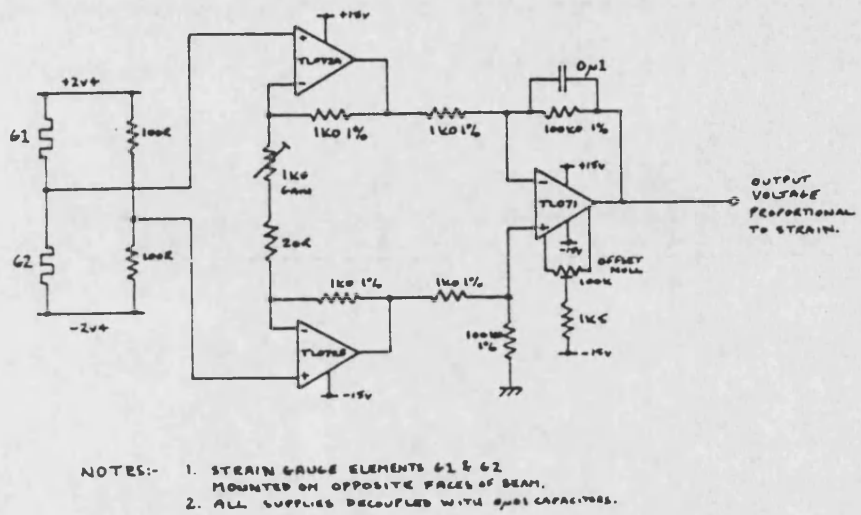


Figure 6.12: Circuit diagram of strain gauge bridge network and amplifier

sociated with its measurement, and as the position computations are based on a series of additions of the deformation measurements, these errors could easily add together to become excessive. Thirdly, the amount of processor time required to make all these measurements and computations could also become rather excessive, leading to inaccuracies due to the time delays involved. This latter difficulty may not be important if the information is only to be used for off-line analysis, and not for real-time applications.

6.5 Computing System

The heart of the experimental rig is a multiprocessor digital computing system. The development of systems of this form has been carried out over a period of several years at the School of Electrical Engineering at the University of Bath, for use in real-time simulation and control problems. In the following sections which describe the hardware components of this system, only the portions specific to this application are described in detail, the remaining parts are only described briefly, as their development and function is described in detail elsewhere.

The next section describe those parts of the hardware which have been

subject to the development indicated above, and the following section describes the parts of the hardware which are specific to this application.

6.5.1 Application Independent Hardware

6.5.1.1 Single Board Computers

The processing elements are based on single board computers, the development of which is detailed in [98], thus only a brief description of its facilities and function is given here.

The single board computer was designed as a universal processing node suitable for a range of uses. The primary applications for which the board was designed are :-

1. Operation in a standalone mode as a single or multi-user computing facility. The board supplies the hardware necessary to run a general purpose disc-based operating system, such as UNIX, or TRIPOS.

2. An expanded system with additional input-output capability and, if required, additional main memory. High resolution graphics and local area network interfaces are examples of possible expansions to workstation level systems.

3. Expanded processing system with several single board computers configured as a shared memory multi-processor with input-output compatibility with 1 above. A flexible architecture can be adopted to enable development of multi-processor operating systems for different tasks.

4. Operation as a control system element. This application places different requirements on input-output, and on supervisory control of the board.

The processor used in the single board computer may be either the MC68000 or the MC68010 [99]. The latter supports virtual memory and virtual machine operation. Although the board has been designed to use the 12.5 MHz devices, slower versions may be accommodated by link options. The on-board memory was designed to run at full speed with either of the two processors at 12.5 MHz so that maximum throughput could be achieved. Three memory options are available, 1/4 Mbyte using thirty-two 64Kbit DRAMs, 1/2 Mbyte using sixteen 256Kbit DRAMs, or 1 Mbyte using thirty-two 256Kbit DRAM devices. There is provision for two MC68451 memory

management units for use in virtual and memory managed operating systems such as UNIX. 8Kbytes of EPROM are provided for boot-strapping and diagnostic software, and memory expansion, including EPROM expansion, is available via the global bus.

Two S6551 programmable baud rate serial interface devices provide RS232 standard serial input-output with optional modem control signals. The S6551 devices are connected to the single board computer interrupt bus to allow interrupt driven operation.

Parallel input-output and the system timer function is provided by the MC68230 programmable interface timer. The timer and input-output sections are logically separate and have separate vectored interrupts. The input-output section also has a direct memory access request output and may be configured as a SASI winchester disc interface by suitable link options.

Interface to floppy disc drives may be made by an optional WD279X-02 floppy disc controller/formatter and support logic. The floppy disc interface provides connection for up to four double sided drives with programmable density, step rates, etc. The interface provides vectored interrupts and a direct memory access request.

A HD68450 four channel direct memory access controller may be used for fast data transfers between memory and input-output devices. On the single board computer one channel has been reserved for the floppy disc unit and a second channel for the SASI interface with the remaining channels used for memory block moves or off-board input-output devices.

Expansion of input-output, memory and processing resources is achieved by the VME compatible global bus connection. The interface circuitry arbitrates for the bus on off-board accesses, controls access from the bus onto the board, and routes interrupts from external input-output devices to the local processor. The circuitry also provides the system clock, system reset, and bus time-out signals.

In this particular application, 8MHz MC68000 devices were utilised, with the master single board computer having 1Mbyte of DRAM, and the two slaves having 1/4Mbyte of DRAM each. Memory management units were not required as the operating system supported is TRIPOS [100], which is not a virtual memory operating system, but which has been extended to suit to the multiprocessor enviroment [98].

6.5.1.2 Bus Arbitration Card

The bus arbitration card is a state machine [97] which ensures that requests for the use of the backplane bus by the single board computers are controlled so that only one single board computer can access the backplane bus at any one time, thus avoiding the possibly catastrophic "deadly embrace" should two single board computers attempt to access the bus simultaneously [101]. The development of this card is described in [98], so only a basic outline of its operation is now described.

When a single board computer wants to use the backplane bus it issues a "bus request" signal to the arbitration card, and waits for a "bus grant acknowledge" signal to be returned from the arbitration card. The arbitration card checks the priority level of any pending "bus request" signals and issues a "bus grant acknowledge" signal to the waiting single board computer with the highest priority. When a single board computer has finished using the backplane bus, the arbitration card again checks for any pending "bus request" signals, and issues a "bus grant acknowledge" signal, if appropriate.

6.5.1.3 Graphics Card

The development of the colour graphics card, referred to as the EFCIS card, has been detailed in [98], so only a basic description is given here. The card is based on the Thompson EF9366 colour graphics controller which features 2 pages of 512 x 512 pixels in 8 colours. The paging facility allows flicker free moving displays to be produced by invisibly refreshing the non-displayed page. The graphics card appears in the input-output page of the memory map, so any of the single board computers can drive it. Displays are generated by sending instructions and data to the graphics card, which can be done directly, but in cases where run-time is not important it is easier and software development is quicker with the use of the Graphics Kernel System [102] software (see Chapter 7), which provides high-level graphics commands for producing lines, characters, etc.

6.5.1.4 Panel Display Card

The panel display card monitors the backplane bus, buffering all the signals and using them to drive light emitting diodes. This makes it possible to get a visual impression of the backplane activity, as the address bus, data bus, and all the control lines can be observed in action. Obviously this is not particularly useful during normal operation, but it is very useful for

debugging purposes, especially when debugging input-output hardware and software.

6.5.1.5 Memory Card

The memory card on the backplane is based on 64Kbit × 1 DRAM devices and provides 1/2Mbyte of additional memory. The refresh circuitry is provided on-board, so the card functions as a stand-alone device. Again, this memory appears directly in the system's memory map, so it can be acessed by any single board computer.

6.5.1.6 Disc Drives

A 40Mbyte Winchester disc provides permanent data storage. The single physical disc is configured as four logical discs each of 10Mbyte capacity. One logical disc is reserved as a system disc, whilst the remaining three are used for program and data storage.

Two $5\frac{1}{4}$ inch flexible disc drives provide additional permanent storage capacity, with each formatted disc having a capacity of about 800Kbytes.

6.5.1.7 Terminal

A BBC microcomputer and a monochrome monitor are used as a terminal. Firmware developed at the University of Bath is used to control the BBC microcomputer such that it behaves either as the usual alphanumeric type of terminal, or as a Tektronix terminal emulator. The latter facility can be used to display graphs on the monochrome screen with higher resolution than that obtainable with the EFCIS card and its colour monitor. In addition, the BBC microcomputer firmware enables screen dumps to be made to an Epsom compatible printer connected to the BBC microcomputer's parallel printer port. This facility is useful to obtain hard copies of graphs without the time-consuming file transfer to the mainframe and subsequent re-plotting on the barrel plotter, although the plots obtained on the Epsom printer are of much poorer quality than those obtained on the Benson barrel plotter.

6.5.2 Application Specific Hardware

6.5.2.1 ADC/DAC Card

The conversion of analogue voltages to digital data is carried out using a 12 bit analogue to digital converter (ADC). The input to the converter has to

be held constant during the conversion, and this is achieved using a "sample and hold" device.

The interfacing of ADC's to microprocessors is often done using interrupts, where the processor instructs the converter to begin its conversion and continues doing other tasks, or idles, until the converter issues an interrupt to the processor to indicate that the data is available. Thus the software has to be written in such a manner as to allow the processor to carry out other processing rather than idle, because this would obviously reduce its overall processing throughput. Although this can easily be achieved within a multi-tasking enviroment, an alternative interfacing technique has been used here which utilises some simple stand-alone synchronous logic to control the conversion process, and provides a current data word that the processor can access at will. Thus the operation of the conversion process is autonomous to the processor, enabling much simpler software to be utilised, and also avoiding the problem of having to handle multiple conversion requests which could occur as any processor can access the converter card. With the interfacing technique used here, the output of the converter appears to any processor in the system simply as a memory location, so multiple data requests are dealt with automatically by the bus arbitration circuitry in the manner already described. Hence the additional minor complication of a handful of standard m.s.i. logic devices places less demands on the software, offering much greater flexibility, and completely removing the possibility of idle states. The operation of the conversion process is now described.

Referring to the circuit diagram shown in figure 6.13, the analogue signal to be converted is initially passed through a second order Butterworth filter to minimise aliasing effects. The filter is constructed around a bi-fet operational amplifier (TL071), and has a cut-off frequency of about 100Hz.

The filtered signal is then passed to the input of a sample and hold device (LF398), which holds the voltage at its output constant when commanded to do so. This held voltage is then converted to a 12 bit digital word by a 12 bit successive approximation ADC (574). The converter is configured for bipolar operation with an input range of ± 10.0 Volts.

The output of the converter is latched into the lower 12 bits of a 16 bit data latch ($2\times$74LS374) which is used to hold the data word for accessing by a processor. As has already been mentioned, the complete conversion process is controlled locally by a handful of m.s.i. logic devices so that the converter operation is independent of the rest of the system.

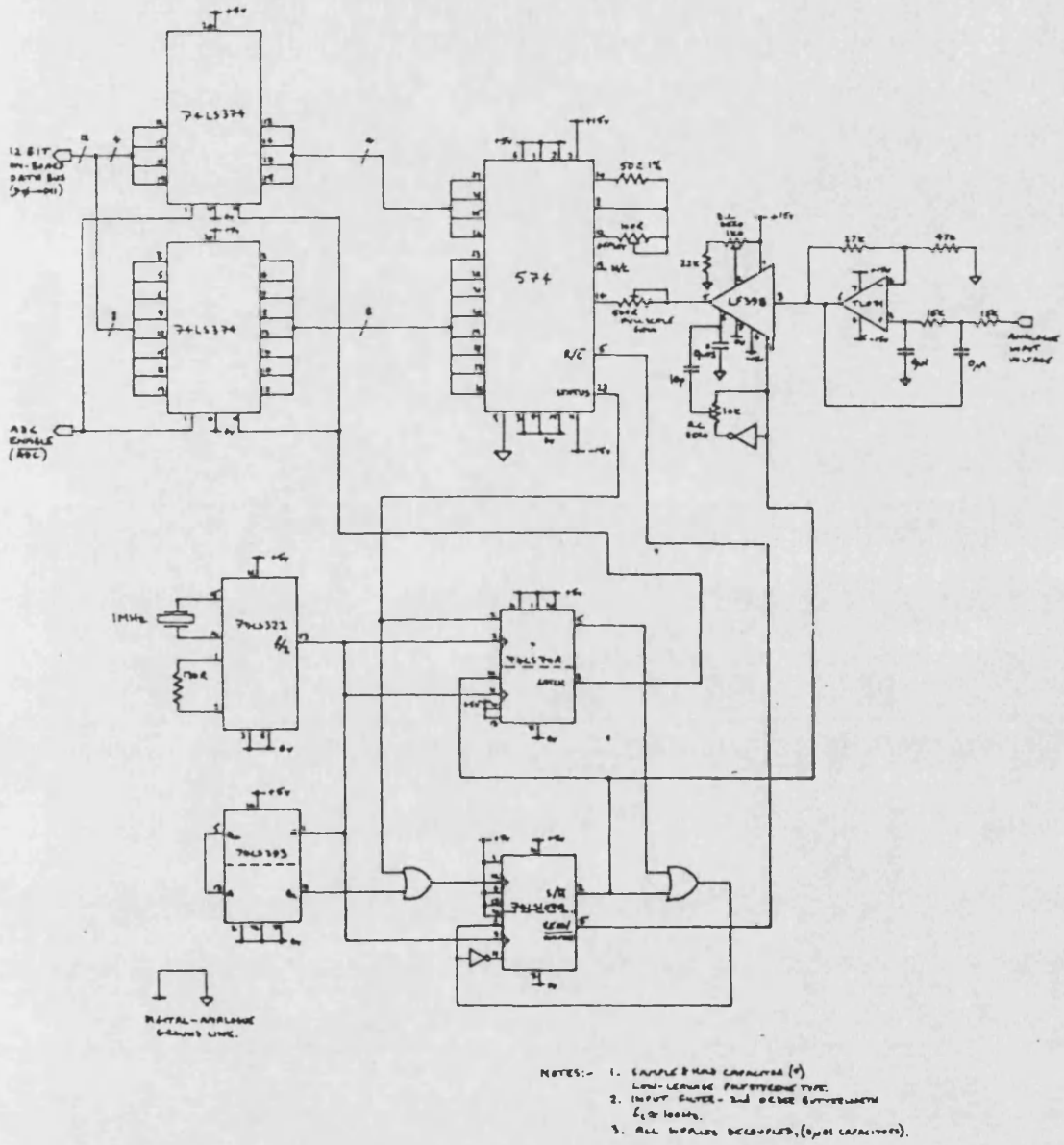


Figure 6.13: Circuit diagram of analogue to digital converter

With reference to the timing diagram shown in figure 6.14, the operation of this configuration is now described. A crystal controlled oscillator (74LS321) with a 1.0MHz crystal provides a stable clock signal. The $f/2$ output of the oscillator is utilised which provides a 500KHz signal, which is used to clock the various flip-flops. The 500KHz signal is divided by 64 using a dual 4 bit binary counter (74LS393) to produce a 7.8125KHz signal. The negative edge of this 7.8125KHz signal causes the JK type flip-flop (1/2 of a 74LS107A) to change state. The output of this flip-flop is used to force the sample and hold device (LF398) into its "hold" mode. On the next positive edge of the 500KHz clock, the $LATCH$ signal goes low, although this causes no action.

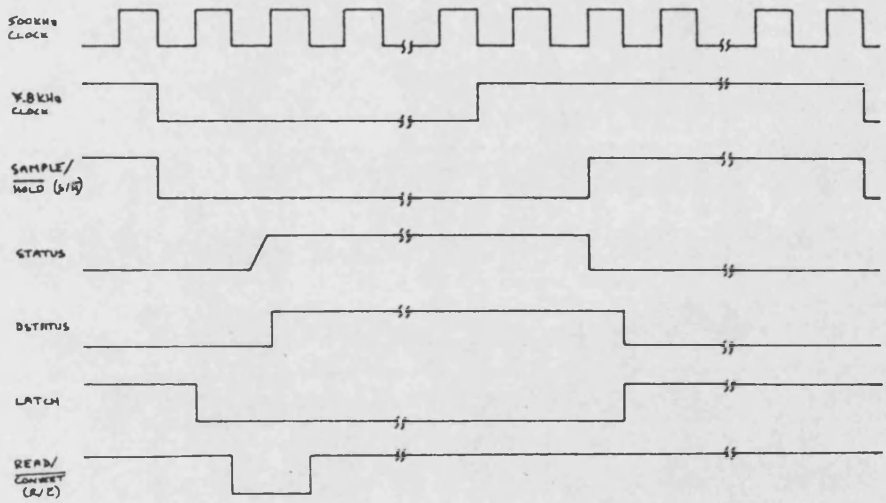


Figure 6.14: Timing diagram for analogue to digital converter control

At the next negative edge of the 500KHz clock, the other JK type flip-flop (1/2 of a 74LS107A configured as a D type flip-flop) changes state, causing the $READ/\overline{CONVERT}$ line to go low, which forces the ADC to begin conversion of the signal at its input, currently being held constant by the sample and hold device. The $2\mu s$ delay between the *hold* command and the *convert* command allows the output of the sample and hold device to settle before conversion takes place. When the conversion has been initiated, the *status* output of the converter goes high. At the next negative edge of the 500KHz clock, the JK flip-flop configured as a D type changes state, returning the $READ/\overline{CONVERT}$ signal to a high level, although this has

no effect on the conversion.

When the conversion has been completed (which takes about 25μs), the *status* output of the converter goes low, indicating that the converted data is available at its outputs. This causes the JK flip-flop to change state releasing the sample and hold device to track the filtered input again. At the next positive edge of the 500KHz clock, the $LATCH$ signal goes high, latching the output data from the converter into the 12 bit latch. The next negative edge of the 7.8125KHz clock causes the procedure to be repeated, *ad infinitum.*

The 7.8125KHz clock represents a period of 128μs, so the absolute maximum delay between the input data and the processor reading the corresponding digital data is 128μs.

The conversion of digital data to analogue signals for driving actuators is carried out using a 12 bit digital to analogue converter (DAC). The circuit diagram detailing the configuation of the DAC's used is shown in figure 6.15. The particular converters used are 12 bit multiplying converters (7545) which are configured to operate in bipolar mode using 2's complement code. The output is amplified by a pair of high performance operational amplifiers (2×OP07) which uses a 10.24 Volt precision reference voltage (LH0071-OH) to give an output voltage range of ±10.24 Volts, which corresponds to a conversion factor of 5mv/bit.

The analogue output signal is subsequently passed through a second order Butterworth filter to minimise aliasing effects. The filter is constructed around a bi-fet operational amplifier (TL071), and has a cut-off frequency of about 100Hz.

The interface logic for this card is now explained with reference to the circuit diagram shown in figure 6.16.

The address lines A_4 through A_{11} are buffered (74LS244) and fed to an eight bit comparator (74LS2521) which is enabled only when both the address strobe line ($\overline{AS}$) and the input-output page line ($\overline{IOP}$) go low, which occurs when a processor declares the address to be valid, and when the address corresponds to a location within the input-output page. The address on lines A_4 to A_{11} is compared to the eight way DIL switch defining the location of the ADC/DAC board in the input-output page of the memory map of the system, and if the comparison produces a true result, the output of the comparator (pin 19) goes low, indicating a successful decode.

The combination of this successful decode signal and the processor pulling either of the data strobe signals ($\overline{UDS}$ or $\overline{LDS}$) low, which it does to indicate its intention to transfer data over the data bus (either a *read* or a

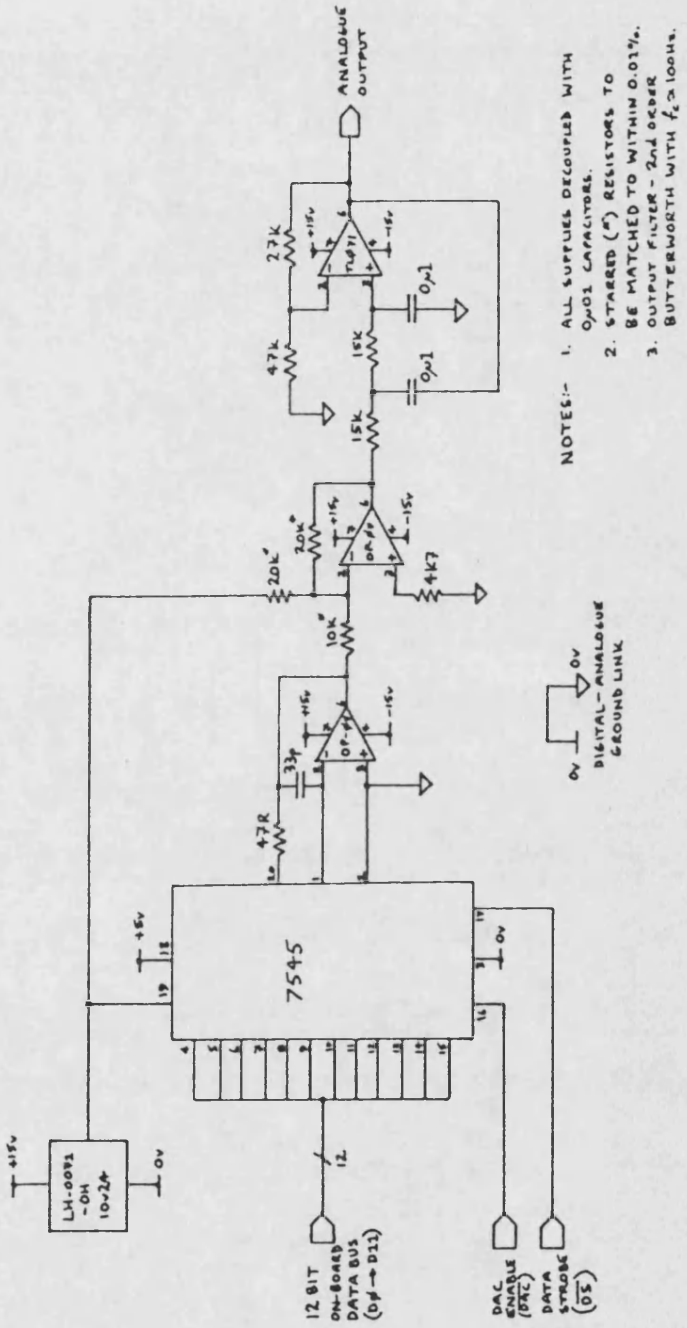


Figure 6.15: Circuit diagram of digital to analogue converter

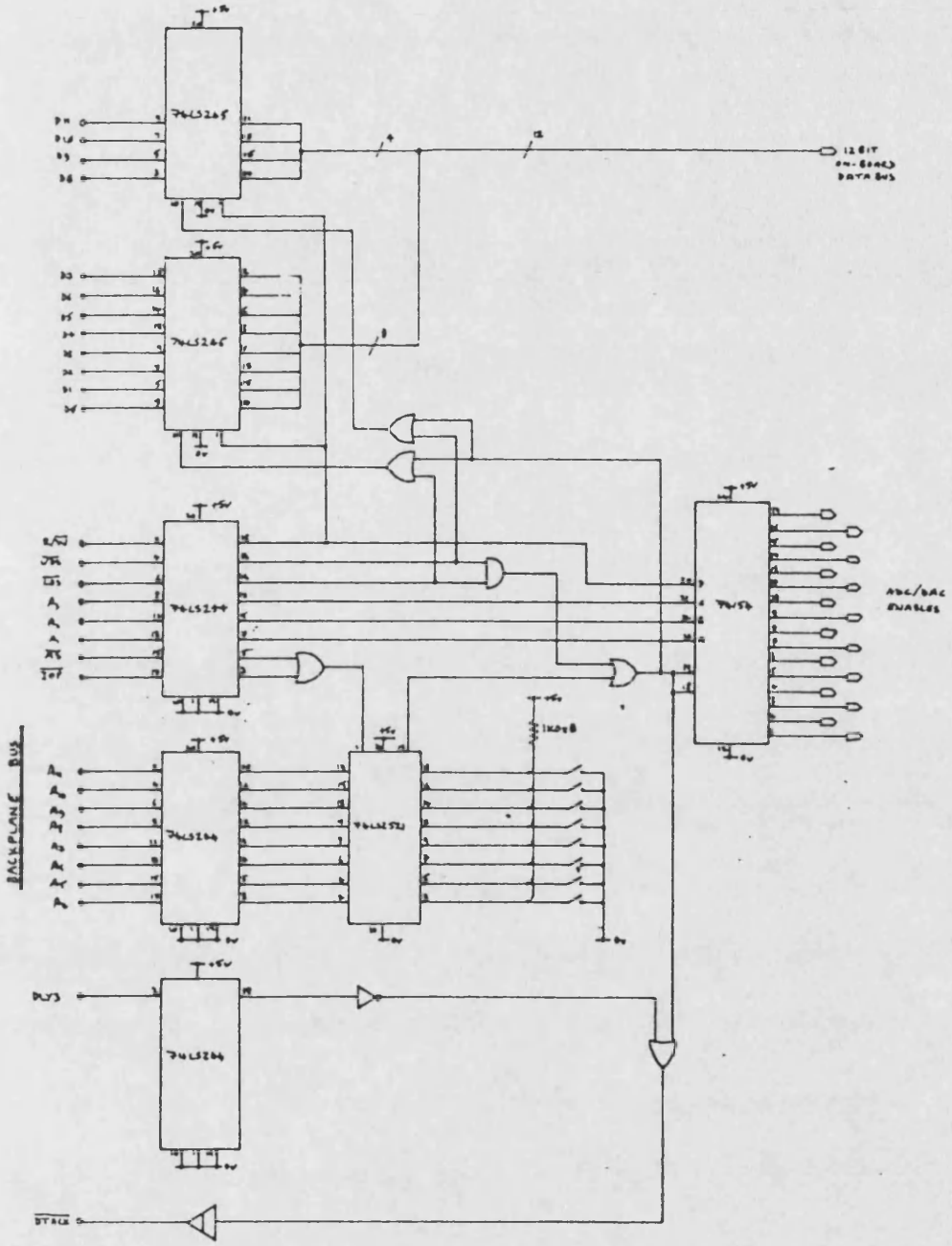


Figure 6.16: Circuit diagram of interface logic for ADC/DAC card

write transfer), causes the 4 to 16 line decoder to be enabled. This decoder pulls one of its sixteen lines low depending on the state of its four inputs, A_1,A_2,A_3, and $R/\overline{W}$. The three address lines define eight physical locations within the input-output page of the memory map, and the $R/\overline{W}$ line defines whether these locations are to be read from, or written to. Thus one of eight outputs of the decoder is pulled low when the processor is executing a *read* cycle, and one of the other eight outputs is pulled low if the processor is executing a *write* cycle. Obviously, the outputs corresponding to read locations are used to enable the outputs of the latch holding the result of the analogue to digital conversion, whereas the outputs corresponding to the write locations are used to enable the inputs of the digital to analogue converters.

The signal used to enable the 4 to 16 line decoder and the data strobe signals ($\overline{UDS}$ and $\overline{LDS}$) are also used to enable the bidirectional buffers connecting the board data bus to the backplane data bus. The direction of the buffers (that is enabling data transfer either onto the board, or off the board) is controlled by the $R/\overline{W}$ signal, obviously such that a *read* signal causes the "off-board" buffers to be enabled, and a *write* signal causes the "onto-board" buffers to be enabled.

In order to indicate to the processor that the address decoding has been successful, and that a data transfer should proceed, the signal which is used to enable the 4 to 16 line decoder (which indicates a successful decode) is used to pull the data acknowledge line ($\overline{DTACK}$) low after a suitable delay time to ensure the data has settled on the data bus. This delay is introduced by the DLY3 signal which is effectively the inverse of the address strobe signal ($\overline{AS}$) delayed by three clock cycles. Thus the address decode logic has up to three clock cycles to decode the address, enable the data bus buffers, and allow the data to settle before the data transfer takes place.

When the processor has completed the data transfer, all the address lines and control lines are released, consequently returning the decode logic to its initial state.

6.5.2.2 Transducer Processing Card

The function of the ultrasonic transducer processing circuits have already been described in Section 6.4.2.1, and will not be re-iterated here. Eight individual processing circuits are built on this card, and obviously some form of interface to the backplane bus is required, and this is now described.

The interface requirements of the ultrasonic transducer card are essen-

tially similar to those of the ADC/DAC card, except that only *read* data transfers are required. Referring to the circuit diagram shown in figure 6.17, as for the ADC/DAC card, the address lines A_4 through A_{11} are compared with the values specified by the DIL switch positions when the address strobe signal ($\overline{AS}$) and the input-output page signal ($\overline{IOP}$) indicate that a valid address in the input-output page has appeared on the address bus. If the comparison is successful, the output of the comparator goes low, which causes the 3 to 8 line decoder (74LS138) to be enabled, but only if the processor is on a *read* cycle, that is if the read/write signal ($R/\overline{W}$) is high. When the 3 to 8 line decoder is enabled, it pulls one of its eight output lines low corresponding to the level of its inputs, address lines A_1, A_2, and A_3. When an output of the 3 to 8 line decoder goes low, the outputs of the latches holding the result of the appropriate transducer circuitry are enabled, putting their data onto the on-board data bus.

The sucessful decode signal is combined with the data strobe signals ($\overline{UDS}$ and $\overline{LDS}$) to enable the data bus buffers, thus connecting the on-board data bus to the backplane data bus. The successful decode signal is also used to pull the data acknowledge signal ($\overline{DTACK}$) low when both the data strobe signals are low, indicating to the processor that the data is available on the bus. Note that no delay is introduced here, as for the ADC/DAC interface, because the 74LS374 and 74LS244 buffers have very short response times so the processor does not have to wait for the data to settle.

When the processor completes the *read* cycle, it releases the address lines and the control lines, causing the decode logic to return to its initial state.

6.5.3 Real-Time Control Requirements

In order to specify the computing requirements, a simple analysis was carried out to predict the computational time required for the implementation of various types of dynamic controller. By far the most significant usage of processor time in the implementation of a control law is associated with the actual arithmetic computations involved, and thus the following analysis considers primarily the arithmetic operations. It should be noted that the analysis is based on the discrete state equation representation of dynamic systems, and also that the symbols α and β are used to represent the time required for a multiplication operation and an addition operation, respectively. The symbols n,m, and p represent the order of the dynamic controller, the number of outputs from the system (number of inputs to the

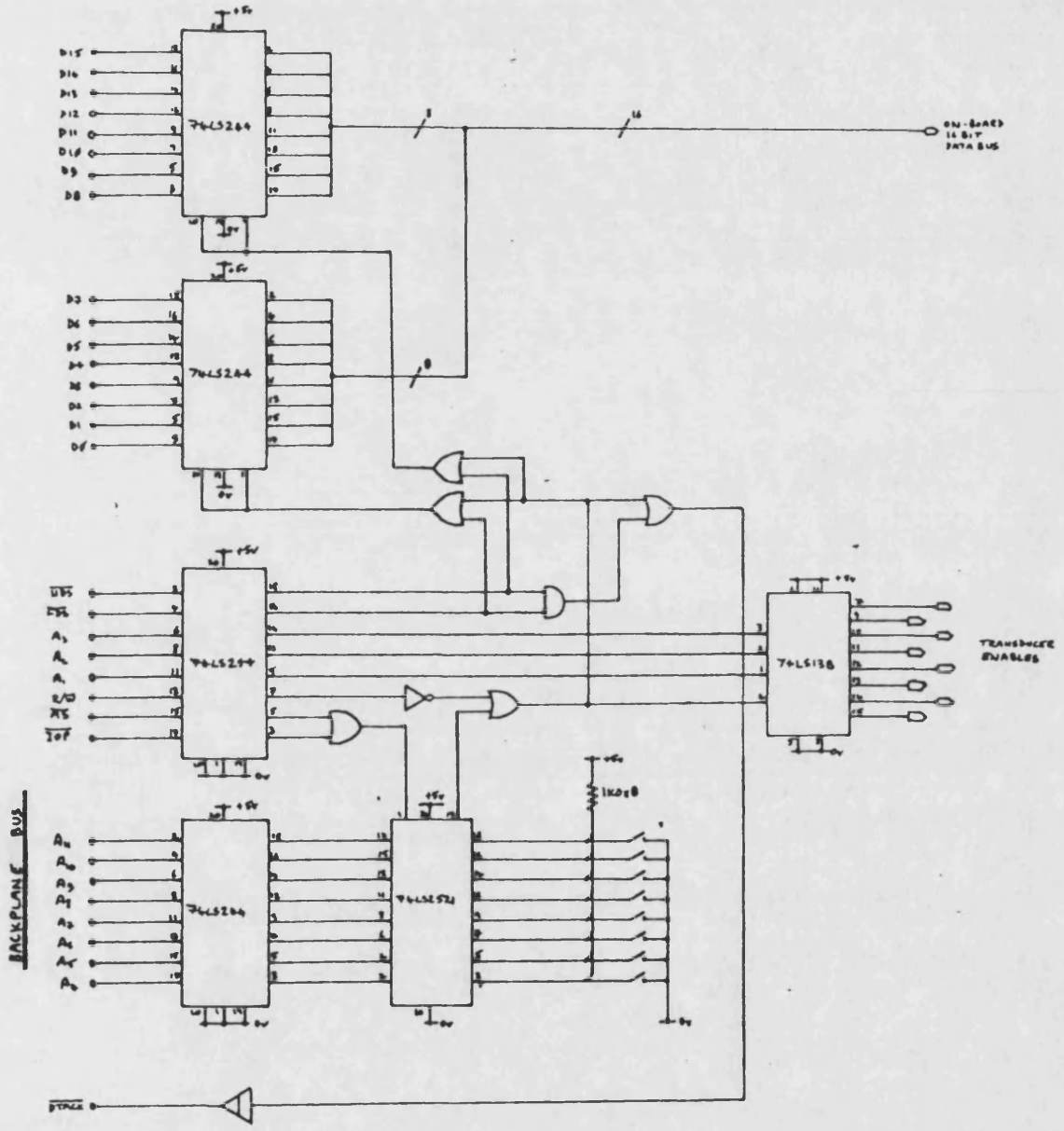


Figure 6.17: Circuit diagram of ultrasonic transducer card interface logic

controller), and the number of inputs to the system (number of outputs of the controller), respectively.

6.5.3.1 Full Order State Estimator

The equations which have to be solved in the case of the full order state estimator are as follows;-

$$\begin{aligned} \hat{x}_{k+1} &= \Phi\hat{x}_k + \Theta u_k + H\left(y_k - C\hat{x}_k\right) \\ u_{k+1} &= -F\hat{x}_{k+1} \end{aligned}$$

The total time required to compute the next feedback term u_{k+1} is given by;-

$$\begin{aligned} T &= 2n\beta + n^2(\alpha+\beta) + np(\alpha+\beta) + nm(\alpha+\beta) + m\beta + mn(\alpha+\beta) \\ &\quad + pn(\alpha+\beta) \\ &= \alpha\left(n^2 + 2np + 2nm\right) + \beta\left(n^2 + 2n + 2np + m + 2nm\right) \end{aligned}$$

It is interesting to note that when $n \gg m, p$, the n^2 terms will dominate the expression, and thus;-

$$T_{n \gg m,p} = n^2\left(\alpha + \beta\right)$$

6.5.3.2 Minimal Order State Estimator

For the case of a discrete time minimal order state estimator, the equations to be solved are;-

$$\begin{aligned} z_{k+1} &= \Phi z_k + \Theta u_k + \Gamma y_k \\ \hat{x}_{k+1} &= K_1 z_{k+1} + K_2 y_{k+1} \\ u_{k+1} &= -F\hat{x}_{k+1} \end{aligned}$$

The total time required to compute the next feedback term u_{k+1} is given by;-

$$\begin{aligned} T &= 2q\beta + qm(\alpha+\beta) + qp(\alpha+\beta) + q^2(\alpha+\beta) + n\beta + nq(\alpha+\beta) \\ &\quad + n(n\alpha + m\beta) + np(\alpha+\beta) \\ &= \alpha\left(n^2 + q^2 + nq + pq + mq + np\right) \\ &\quad + \beta\left(q^2 + 2q + n + qm + qp + nq + nm + mp\right) \end{aligned}$$

where q is the order of the estimator, that is $q = n - m$ (see Chapter 5). Substituting for q gives;-

$$\begin{aligned} T &= \alpha\left(3n^2 + m^2 - 2nm - 2mp + 2np\right) \\ &\quad + \beta\left(2n^2 - nm + 3n - 2m + 2np - mp\right) \end{aligned}$$

As in the previous section, it is interesting to note that when $n \gg m, p$, the n^2 terms dominate the expression for T. Thus;-

$$T_{n \gg m,p} = 3n^2\alpha + 2n^2\beta$$

Note that the expression above will always be greater than the expression given for T when $n \gg m, p$ for the full order state estimator as discussed in the previous section. However, when n is of similar order as m and p, the minimal order state estimator can offer a computational advantage. As an example, consider the case when $n = 5$, $m = 2$, $p = 1$. The computational time for the full order state estimator is given by;-

$$T_{fo} = 55\alpha + 67\beta$$

and the computational time for the minimal order state estimator is given by;-

$$T_{mo} = 65\alpha + 59\beta$$

Taking the difference gives;-

$$\delta T = T_{fo} - T_{mo} = -10\alpha + 8\beta$$

For 32 bit floating-point operations on the single board computers used in the rig, $\beta \simeq 1.67\alpha$, hence;-

$$\delta T = -10\alpha + 13.42\alpha = 3.42\alpha$$

Hence for this case, the minimal order state estimator would take 3.42α less time to compute than the full order state estimator.

6.5.3.3 Kalman Filter

The steady-state implementation of a Kalman filter (where the gain matrix is constant and corresponds to the steady-state solution of the associated Riccati equation, see Chapter 5), obviously has the same computational

requirements as the full order state estimator, as discussed in Section 6.5.3.1. However, the full on-line implementation of a Kalman filter requires the computation of various covariances which lead to the computation of the gain matrix itself. Thus, in addition to the equations given in Section 6.5.3.1, the following discrete time equations also have to be solved;-

$$\begin{aligned} P'_{k+1} &= \Phi P_k \Phi^T + \Theta Q \Theta^T \\ K_{k+1} &= P'_{k+1} C^T \left[C P'_{k+1} C^T + R \right]^{-1} \\ P_{k+1} &= \left[I - K_{k+1} C \right] P'_{k+1} \end{aligned}$$

This corresponds to an additional computational time of;-

$$\begin{aligned} T_{add} &= 2n^2(\alpha+\beta) + 2pn(\alpha+\beta) + nm(\alpha+\beta) + m^2(\alpha+\beta) \\ &\quad + m^3\alpha + n^2\beta + 2nm(\alpha+\beta) \\ &= \alpha\left(2n^2 + 2pn + 3nm + m^2 + m^3\right) \\ &\quad + \beta\left(3n^2 + 2pn + 3nm + m^2\right) \end{aligned}$$

where it has been assumed that the time required to compute the inverse is approximately $m^3\alpha$. Thus the total time required to compute the next feedback term u_{k+1} is given by;-

$$T = \alpha\left(3n^2 + 4np + 5nm + m^2 + m^3\right) + \beta\left(4n^2 + 2n + 4np + 5nm + m^2\right)$$

Again, it is interesting to note that when $n \gg m, p$, the n^2 terms will again dominate the expression, and thus;-

$$T_{n \gg m,p} = 3n^2\alpha + 4n^2\beta$$

As would be expected, this computational time is always greater than that required for an equivalent fixed gain state estimator.

6.5.3.4 Requirements For Beam

The implementation of a controller based on a 6th order fixed gain state estimator and a corresponding state feedback law for the experimental beam is now considered. The assumption of a 6th order state estimator is to allow for control of three modes of the beam, one rigid mode and two elastic modes. It will also be assumed that two measurements are available (angular

position and rate at the pivot point) and that one input is required (torque actuator acting about the pivot point). This situation has been chosen for its similarity to a single axis attitude control problem. Thus, $n = 6$, $m = 2$, $p = 1$, and the time required to compute the next feedback term (from Section 6.5.3.1) is;-

$$T_{step} = 72\alpha + 86\beta$$

For 32 bit floating-point calculations using the single board computers, $\beta \simeq 1.67\alpha$, hence;-

$$T_{step} = 215.62\alpha$$

As the cut-off frequency of the anti-aliasing filters is approximately 100Hz (see Section 6.5.2.1), the highest frequency for consideration will be taken to be 100Hz. The computational problem will now be considered as a distributed process, with the computation being shared equally over a number of processors, denoted by x. A factor of 5% will be allowed for inter-processor communications, and thus the actual time taken to compute the next feedback term will be;-

$$T_{actual} = \frac{215.62 \times 1.05\alpha}{x} \text{ secs}$$

For a full speed MC68000 with a clock frequency of 12.5MHz, $\alpha = 49\mu s$, therefore;-

$$T_{actual} = \frac{0.01109}{x} \text{ secs}$$

For the maximum frequency of 100Hz, the number of samples per cycle is given by;-

$$\text{samples per cycle} = \frac{1}{f_{max} \times T_{actual}} = \frac{x}{1.109}$$

Therefore, with regard to the well-known Sampling Theorem (see for example [103],[13]), and allowing a margin of safety of two, the minimum number of single board computers required is given by;-

$$x = 1.109 \times 4 = 4.4$$

That is, five single board computers each running at 12.5MHz will be required.

6.6 Summary

In this chapter the hardware which comprises the experimental rig has been examined. The details of the experimental structure have been discussed and the various constraints associated with its construction have been highlighted. These constraints led to the particular form in which it now exists. The facility to add masses at particular points on the structure allows the dynamic properties to be altered, which may be used to examine the sensitivity of feedback systems to parameter variation or modelling errors.

From the discussion of the various types of actuator that have been considered, it will be apparent that this is still a significant problem area in the further development of the rig. It is felt that the use of air jet actuators shows considerable promise, although the procurement of the necessary jets and valves may prove to be rather expensive, and a suitable compressed air supply would also have to be made available.

Of the two methods considered for measuring the position of the structure at several points, the ultrasonic position transducers have proved to be most useful as they are fast, have a good resolution and range, and they are comparatively cheap to construct, the component costs being around £20 per transducer.

The hardware aspects of the computing system have been discussed, although only the components that have been specifically developed for this rig were examined in detail, these components being essentially input or output devices.

Finally, the processing requirements for real-time control have been examined. A particularly interesting fact came to light regarding the comparison of the computational aspects of minimal order state estimators and their equivalent full order state estimators in that it is often assumed that the computational requirements of the equivalent minimal order state estimator will be lower than that of the full order state estimator. However, it has been shown that as the system order increases, the time required for the minimal order state estimator increases to match that of the full order state estimator, and then exceeds it. In fact, as the system order becomes much greater than the number of inputs and outputs, the minimal order state estimator requires twice as much computational time as the equivalent full order state estimator.

Due to financial constraints, the processing requirements as identified in Section 6.5.3.4 for real-time control has not been met, the available processing nodes being three single board computers all running at 8.0MHz.

This combination has only 38% of the processing capacity of that required, and thus reduces the possible complexity of any controller that may be implemented. In fact, for the same circumstances of the example discussed in Section 6.5.3.4, the maximum order of the controller that may be implemented is only 3rd order, thus the range of control laws that may be implemented with this configuration is extremely limited. Note also that the timing analysis carried out does not make any provision for processing (such as scaling) of input data from sensors, or output data for actuators, and thus full implementations will almost certainly require additional time for such processing.

Chapter 7

Experimental Rig — Software

7.1 Introduction

In this chapter the software developed to provide various facilities for the experimental rig is described. The first section details the operating system used for the computing system which forms the basis of the experimental rig.

The next section describes two major software utilities, the Kermit file transfer program for communications with other computers via an RS232 serial data link, and the Graphical Kernel System (GKS) which supports a set of high-level graphics commands that are used primarily for graph plotting routines.

The subsequent section describes the manner in which multiprocessor simulations have been constructed using the computing system, with details of the function of various tasks within the simulations, and the software used for off-line display of data files produced by the simulations.

The data files produced by the graph plotting routines can be transferred to the mainframe computer for plotting on a high-resolution barrel plotter, and the graphics processing software developed for the mainframe which is used to achieve this is also described. In addition, the implementation of various control system design algorithms on the mainframe is discussed.

7.2 Operating System For Experimental Rig Computing System

The operating system used for the experimental rig computing system is TRIPOS [100], which is a multitasking real-time operating system designed primarily for minicomputers. The TRIPOS operating system is documented in [104], [105], and [106], so it will not be detailed here. The necessary extensions to TRIPOS to enable it to be used as a multiprocessor operating system are detailed in [98], so again, these will not be described here.

As is normal practice in a multitasking operating system, each task has a priority associated with it which enables the programmer to ensure that a task which should be executed as rapidly as possible has highest priority so that it is executed whenever possible. The operating system incorporates a task scheduler which attempts to run the task with the highest priority which is available to be run at all times. Thus when a task is suspended at any time, the scheduler attempts to identify the task with the highest priority which is waiting to be run, and runs that task until it too suspends, when it again looks for the waiting task with the highest priority, and runs that task until it suspends, and so on, *ad infinitum*. The multiprocessor operating system merely extends this by supporting additional tasks on slave processors, except that these tasks are not subject to the control of the task scheduler, but are free to run independently of tasks on the master processor, and of tasks on other slaves.

Inter-task communication is facilitated by passing "packets" between tasks. A packet contains five fields, a link field, an identification field, a type field, a result field, and an argument field (see figure 7.1). The link field provides a pointer to the next packet on a task's packet queue, and the identification field contains a parameter identifying the task which the packet is bound for, but on passing a packet, the identification field is altered to the sending task's *i.d.* so that the packet can be returned if so desired. The type field contains a parameter which enables the receiving task to identify what it should do with the packet, and likewise for the sending task if the packet is returned. The result field contains two parameters which are only set by the receiving task before the return of a packet to enable the receiving task to return results of processing the packet to the sending task. The argument field contains up to seven parameters which are used by the sending task to pass information to the receiving task for processing.

It should be noted however, that the complete packet is not transferred

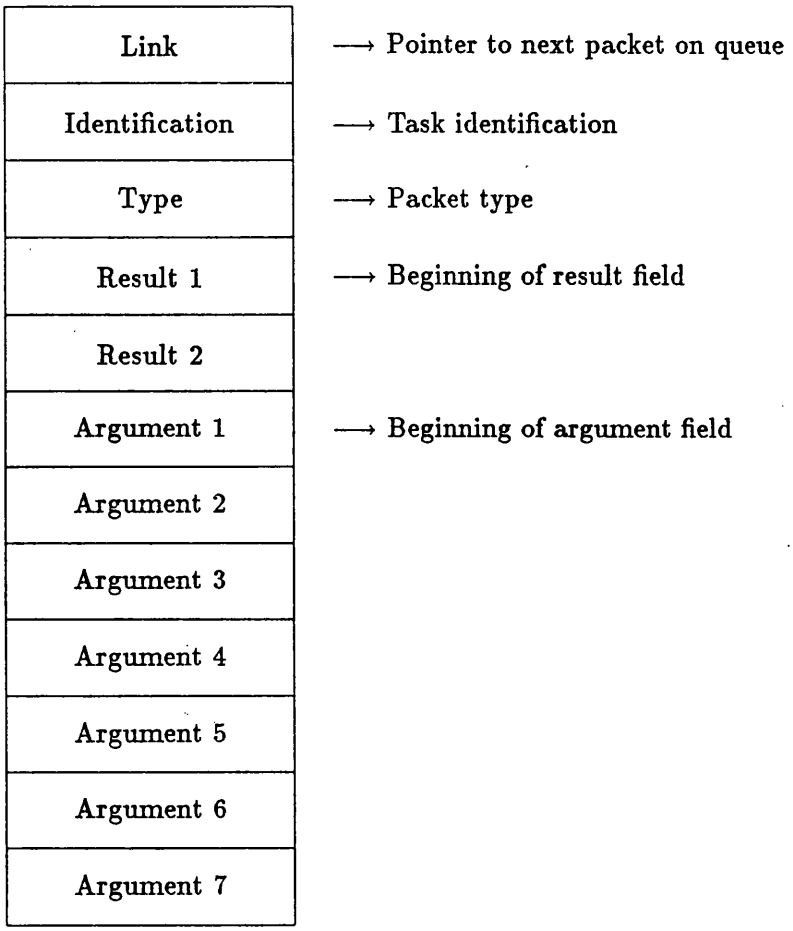


Figure 7.1: Packet data structure

between tasks, only a pointer to the packet is passed, thus the overhead associated with packet communications is very small.

The operating system is written in BCPL [107] (except for the machine dependent parts which are written in MC68000 assembly code [99]), which is a typeless, block-structured systems programming language, which in this implementation, has a floating-point extension. Due to the ease with which operating system functions can be called (such as the packet passing commands), BCPL is also used for the simulation and data-processing software.

The typeless nature of BCPL offers programmers tremendous freedom in program design, allowing the creation of data structures as required, rather than being constrained to the pre-defined data structures used in most programming languages. However, whilst this flexibility offers freedom, it also imposes great responsibilty on the programmer. This is because the freedom implies that the compiler has limited checking that it can carry out, thus it is easy to produce syntatically correct code which the compiler will compile and produce object code for, but which is completely meaningless.

7.3 Utilities For Experimental Rig

Two particularly useful software utilities are the Kermit file transfer program which allows communication with other computing systems, and the GKS graphics routines which allow hardware independent graphics. Both of these are now described.

7.3.1 Kermit

The second serial port on the master single board computer is used to communicate with other computers via an RS232 serial data link. The transfer of files is handled by a program known as Kermit, which is an implementation of a file transfer protocol designed primarily for communications between dissimilar operating systems. A relevant version of the Kermit program is required at both ends of the communication link, but as a very large number of versions of Kermit exist for many different machines, file transfer between a large number of machines is possible.

The details of the protocol are well documented [108], together with instructions for the use of existing versions of Kermit, and guidelines for the development of new versions.

The basic action of the Kermit programs is to transfer data in packets, where the transfer of each packet incorporates its own error checking, such

that Kermit detects the vast majority of failures, and usually manages to recover by re-transmitting the failed packet, up to a user-definable maximum number of times. The rather pedantic nature of Kermit makes it fairly slow, but the ability to transfer data files between vastly dissimilar operating systems more than makes up for this.

The Kermit programs are used for transferring data files from the experimental rig computing system to the mainframe for plotting on a barrel plotter, and also for transferring data files resulting from control system design programs on the mainframe (which are discussed in Section 7.6) to the experimental rig computing system, for example to provide controller parameters for simulations.

7.3.2 Graphical Kernel System

A subset of the Graphical Kernel System (GKS) [102] has been implemented at the University of Bath, and this implementation is available on the experimental rig computing system. The GKS software provides a set of high-level graphics commands, with a variety of colour palettes, and incorporates such features as shading and hidden line removal. However, the overheads associated with GKS slow it down, making it unsuitable for producing real-time animated displays, but it is used as the basis of graph plotting routines where the advantages of high-level commands which improves program development time outweighs the longer run-times. Another advantage that GKS offers is to make programs hardware independent, as different output devices can be used simply by using the appropriate drivers. The graph plotting routines are discussed in more detail in Section 7.5.

7.4 Simulation Software

It has already been mentioned that the experimental rig computing system is used for simulation exercises, and several different types of simulation have been set up. Although the different simulations have their own variations, the structure of all of them is essentially the same, the main differences being associated with the computational tasks running on the slave processors, such as the actual code which is run, and the initialising of these tasks by the monitor task. Thus, rather than describing all the simulations, the type of simulation which has been used most frequently will be taken as an example.

Hence, consider the simulation of a system subject to a state feedback law, where the states are obtained via a state estimator (a particularly relevant example for control of flexible spacecraft). The problem can be organized such that the master single board computer runs three tasks (in addition to the operating system tasks), which are as follows;-

1. A monitor task that provides the user interface to the simulation and is the only task which the user communicates directly with, and which offers various functions such as allowing parameter or input level changes, or logging, displaying, and storing of simulation variables.

2. A simulation control task that basically controls the computations which are carried out on the slave single board computers, and ensures synchronism of the computations, and can produce timing information for real-time simulations.

3. A real-time display task that provides a real-time display on the high-resolution colour graphics monitor of simulation variables. The display can be controlled by the user via the monitor task.

As already mentioned, the computational burden of the simulation is carried by the slave single board computers, and in this case, can easily be split such that one slave single board computer computes the plant states and outputs, and the other computes the estimated states and outputs, with the feedback signal computation being split between them. (See figure 7.2 for a graphical description of this structure.)

7.4.1 Monitor Task

The monitor task acts as an interface between the user and the simulation, thus, when the monitor code is run, the other tasks that form the simulation are set up with the appropriate object code, and initialised. The monitor program can be invoked by entering the command "run-sim", with the appropriate argument string, if required.

When the task structure is complete, the monitor reads the continuous-time state space matrices describing the model, together with a time step for the simulation, from a data file. In this example, the monitor also reads the continuous-time state space matrices defining the state estimator, and the state feedback matrix, together with an integer scaling term. This scaling term allows the time step defined by the model data file to be reduced should the dynamics of the state estimator require a smaller time step. When all

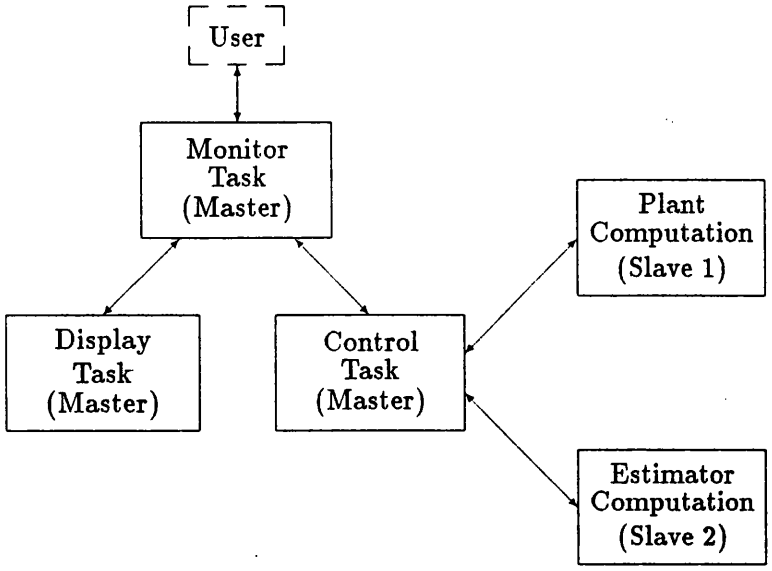


Figure 7.2: Task structure for model/estimator multiprocessor simulation

the data has been read, and the time step defined, the continuous-time state space matrices are converted to their discrete-time counterparts (using the algorithm described in Appendix B.5), which thus defines the simulation equations.

When the discrete-time matrices have been computed, the necessary vectors for the simulation variables are obtained from the freestore management routines and set to zero. At this stage, the data that each task requires is distributed, making the simulation ready to run. Subsequently, the simulation is started and the monitor issues a prompt to the terminal to indicate this.

When the monitor program is invoked, several arguments are available to alter its operation. One of these arguments is a DEBUG switch, which when invoked (the default is off), sends messages to the terminal to indicate the various operations that it is carrying out, and whether they are successful or not. Obviously, the primary use of this is for debugging the simulation code.

Because the model and the state estimator are initialised with zero initial

conditions, neither will alter from this state until perturbed in some manner, and this can be done via the monitor commands. As has already been mentioned, the monitor produces a prompt as soon as the simulation actually starts to run, and then waits for a command from the terminal. A variety of commands are available, and when a command is entered, it is executed, and when completed, the monitor again returns a prompt, and awaits the next command. Any entry at the terminal is compared to the command strings to attempt to identify a match, if a match is not found, then the terminal entry is returned to the screen as part of an error message. Abbreviations in the commands are allowed, but obviously sufficient characters have to be used to make the identification unique, else the wrong command may be invoked. Some commands expect arguments, and the expected arguments can be displayed by typing the command followed by a question mark.

The following commands are available;-

- HELP
- SELECT
- HOLD
- CONTINUE
- SINGLE
- STATUS
- EXAMINE
- DO
- REFERENCE
- PLOT
- STYLE
- SAVE
- RESET
- QUIT

The HELP command merely displays the list of available commands with a short description of the function of each one.

The SELECT command is used to select the required picture to be displayed by the display task. Note that picture number 0 is the simulation overview which is usually displayed, and that selection of picture number -1 causes the display task to be suspended, until it receives a valid picture number request, or a self-destruct request.

The HOLD command suspends the running of the simulation. This is particularly useful in collaboration with other commands which execute code not directly related to the simulation, where the execution of the simulation code would interfere with the subsequent command.

The CONTINUE command is complementary to the HOLD command, and continues the simulation after it has been suspended by the HOLD command.

The SINGLE command forces the simulation into single-step mode. When this command is invoked, one step of the simulation is carried out, and the simulation variables are displayed. The user is prompted for another step to be computed, or if not, whether the simulation should be returned to its continuous mode. This facilty is particularly useful for checking the actual simulation computation code (in this case the model and state estimator code), to ensure that variables are altering as expected.

The STATUS command displays the names and identifying numbers of the tasks associated with the simulation, and whether or not data is being logged from the model or state estimator.

The EXAMINE command displays details of the current simulation regarding the type of model and state estimator, and the time step used in the simulation.

The DO command is used to execute a TRIPOS command string. The string following the DO command is passed to a dynamically created Command Line Interpreter (CLI) task, and executed. This means that all the TRIPOS commands are available to the user whilst the simulation is running.

The REFERENCE command is used to define a new reference input vector to the system. When the command is invoked, the current reference vector is displayed and the user is asked whether it is required to update the vector or not. If it is required to update the vector, the user is then prompted for the new vector, and then asked whether the response of the simulation is to be logged or not. If data logging is not required, the new vector is passed to the model control task where the reference vector is updated at

the next time step (see Section 7.4.2). If data logging is required, the user is prompted for the length of time that the responses should be logged, and the number of steps that should be ignored between logged points. The command then responds with the number of data points that will be logged, and asks the user whether this is satisfactory or not. If the number of data points is not satisfactory, then the selection of time length and points skipped is repeated until a satisfactory number is obtained. At this point, the user is then prompted to identify which simulation variables are required to be logged, and when this is done, the new reference vector is passed to the model control task, where the vector is updated at the next time step, and the data logging is initiated.

The PLOT command is used to plot time histories of the variables logged by the REFERENCE command. This command uses the GKS software (see Section 7.3.2) to plot graphs on either the colour graphics monitor via the EFCIS card, or on the terminal screen (provided the terminal has been set up in Tektronix emulation mode). Low quality hard copies can be obtained of graphs plotted on the terminal screen via an Epson printer by using a screen dump facility within the Tektronix emulation firmware.

The STYLE command is used to alter the manner in which graphs are plotted. Variations such as plot colour (obviously irrelevant for the Tektronix screen), and graph axis grid on or off are possible.

The SAVE command is used to make a permanent copy of data logged by the REFERENCE command on disc. Its single argument is the name of the file to be created containing the logged data. Note that the data is stored in binary for two reasons, firstly to minimise the disc space required, and secondly, to minimise the time needed to write the data to disc, and in subsequent programs, to minimise the time needed to read the data back from disc. The data files have a particular structure, being headed by a file description block, which details the number of time steps at which data was logged, and which variable types are contained in the file (for example, outputs and states). After the file description block, the logged variables can be found in blocks which have a particular order, according to the variable type. The entries in the file description block indicate to a program reading the data file how much memory it should allocate before reading each block. If no variables of a particular type have been logged, then the number of variables in that block will be zero, and the corresponding entry in the file description block will be zero, hence the reading program can ascertain that next value found in the file will correspond to the next non-zero entry in the file description block. A diagrammatical description of this structure is

shown in figure 7.3.

The RESET command is used to reset all the variable vectors in the simulation to zero. This effectively returns the simulation to the same state as when it was started, and is particularly useful to carry out a series of simulations which start from the same initial conditions, using the same simulation data. For example, the step responses of the system for each input channel with the same initial conditions can be simulated without having to exit the simulation program.

The QUIT command is used to terminate the simulation program. It does this by passing packets to all the tasks created by the simulation which instructs them to self-destruct. Note that a self-destruct request is initially lodged with the control task, which only carries out the request at the beginning of the next time step which is a natural break in the inter-task communications. At this point, the slave tasks will also be at a known point (waiting, in fact), thus the monitor task then passes self-destruct requests to the slave tasks, and then similarly for the display task. Finally, the allocated freestore (memory) is released, and then the monitor program is terminated.

The reason for designing the simulations in this manner is two-fold. Firstly, if sufficient computing power is available (or made available at a later date), then this structure allows easy adaptation to a real-time simulator. Secondly, the development of a real-time controller for the experimental rig is likely to proceed along very similar lines, thus code developed for simulations can be easily transferred to the real-time control problem. This then provides a completely safe means of developing control software enabling it to be tested, debugged and verified within a simulation enviroment before attempting to carry out real-time control of the experimental rig.

7.4.2 Control Task

The essential purpose of the control task is to synchronise the operation of the slave tasks so that the computations remain in step. The essential aspects of its operation are quite simple. Once the task has been created and initialised with the data that it requires, it enters a loop which consists of just two major functions;-

- check state of flags
- compute next time step

The first function examines a set of flags, the state of which can be changed via the monitor program commands. For example, if the SINGLE

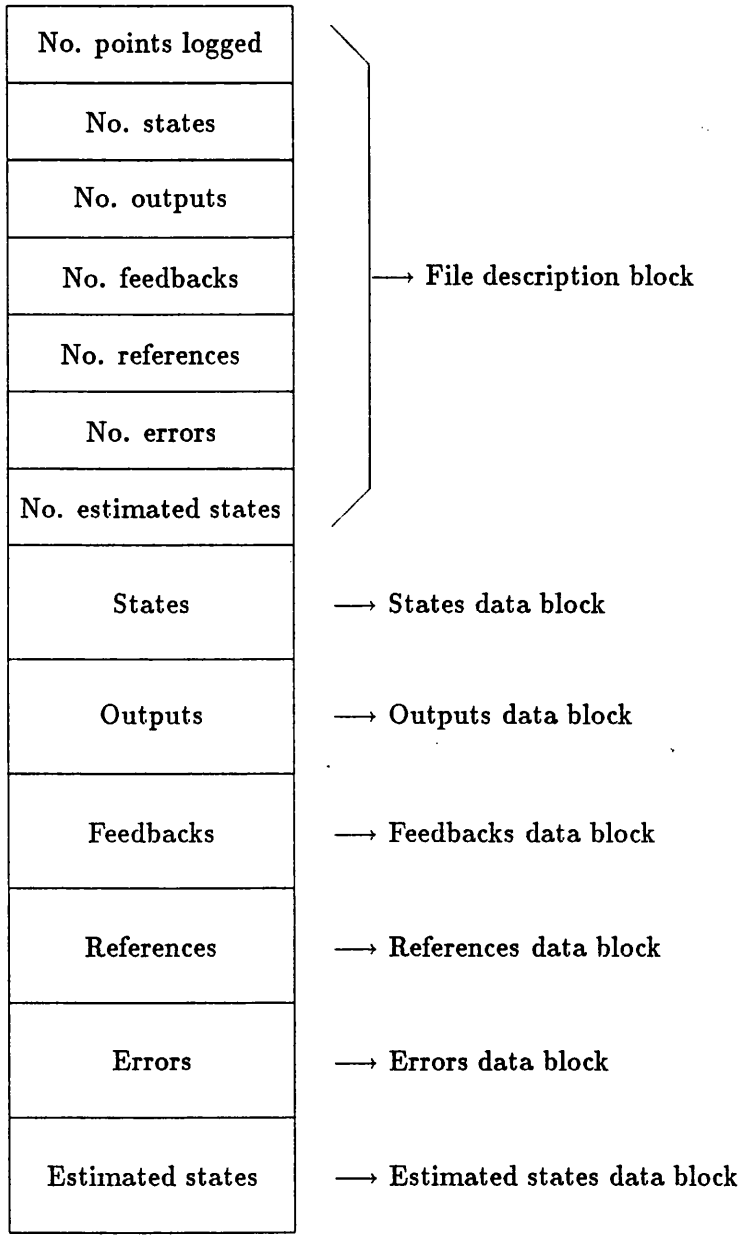


Figure 7.3: Simulation results file data structure

command is entered, the monitor task passes a packet to the control task which sets the single-step flag to TRUE (its normal state is FALSE). When the control task next examines the flags, the change of state of the single-step flag is noted, and the control task then waits for the appropriate packet to be passed from the monitor instructing the control task to compute one step and wait, or return to continuous mode, the latter resetting the single-step flag to FALSE in the process.

The computation of the next time step is carried out by passing packets to the slave tasks as appropriate, which they return when their computations are complete.

The control of the slave tasks by a separate task allows easy re-distribution of the computational load of the simulation, should additional processing nodes become available at a later date, without having to alter any part of the monitor program.

7.4.3 Real-Time Display Task

The real-time graphics display task produces moving pictures using the EFCIS colour graphics board. For general purpose graphics output, this board is normally driven using the Graphical Kernel System (GKS) [102] graphics library. Unfortunately, as already mentioned, GKS incurs large processing overheads which makes it unsuitable for fast moving displays and so the real-time graphics task drives the EFCIS graphics controller directly by sending packets to the controlling driver. This technique, however, does makes the graphics display task hardware dependent.

The features provided by the graphics controller include line and text drawing in eight colours and a paging facility which allows a screen to be invisibly redrawn while another page is displayed. The graphics task uses the paging feature to produce interference-free animated displays and attempts to maximise frame refresh rate by only redrawing the moving picture items.

The main body of the graphics task is a packet handler which accepts requests from the simulation main body while performing the display refresh cycle. The task performs three actions which are;-

- change display
- stop display
- self-destruct

Each display is produced by a general purpose display control routine which acts on the picture data structure for the currently selected display. The picture data structure is composed of a picture table (see figure 7.4), which contains pointers to picture control blocks (see figure 7.5), which in turn contains pointers to work control blocks (see figure 7.6), and pointers to various packet lists defining the necessary actions to produce the picture. The display lists are single linked lists of pre-formed packets which are queued to the graphics driver in order to either draw or undraw the display items. The fixed item list is used to draw the non-moving parts of a picture on both pages. It is only processed when the picture is initialised. List1 and list2 are display lists which draw the moving display items on page1 and page2, respectively. Finally, the work lists contain procedures and data items which are used to move the display items drawn by list1 and list2.

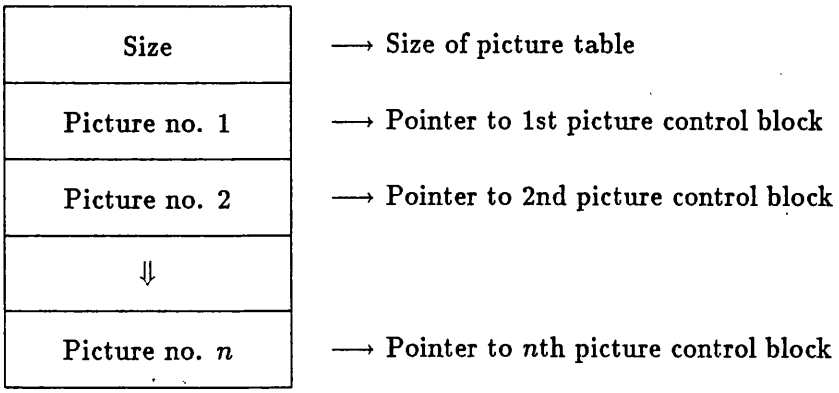


Figure 7.4: Picture table data structure

The graphics initialisation routine generates the picture data structures for each display required in the simulation. Each display is created as a mosaic of standard "meter" panels which are set up by calling a definition procedure. The definition procedure forms the required packets and work list entries and sets up any scaling factors and titles which are specific for that displayed measurement. This structure makes it a simple matter to automate the generation of displays for any simulation study.

The final action of the graphics task initialisation routine is to store the picture identifiers so that the simulation SELECT command can be used to

Frame count	⟶ Number of times picture has been updated
Fixed list	⟶ Pointer to packets to draw non-moving items
List1	⟶ Pointer to page1 packets to draw moving items
Work1	⟶ Pointer to work control blocks for list1
List2	⟶ Pointer to page2 packets to draw moving items
Work2	⟶ Pointer to work control blocks for list2
GETDATA	⟶ Pointer to routine to get data for this picture
Buffer	⟶ Pointer to buffer passed by GETDATA

Figure 7.5: Picture control block data structure

select them.

7.4.4 Model Task

The model task is very simple in operation. After creation and initialisation by the monitor program, the model task waits to be passed a packet asking it to do one of three things;-

- compute the next set of states and outputs of the model
- compute the top half of the feedback vector
- self-destruct

The first two functions are largely self explanatory, as the fact that the discrete state equations are used for the simulations has already been mentioned, so the steps required to compute the next set of states and outputs are obvious, and similarly with the feedback vector. On receipt of

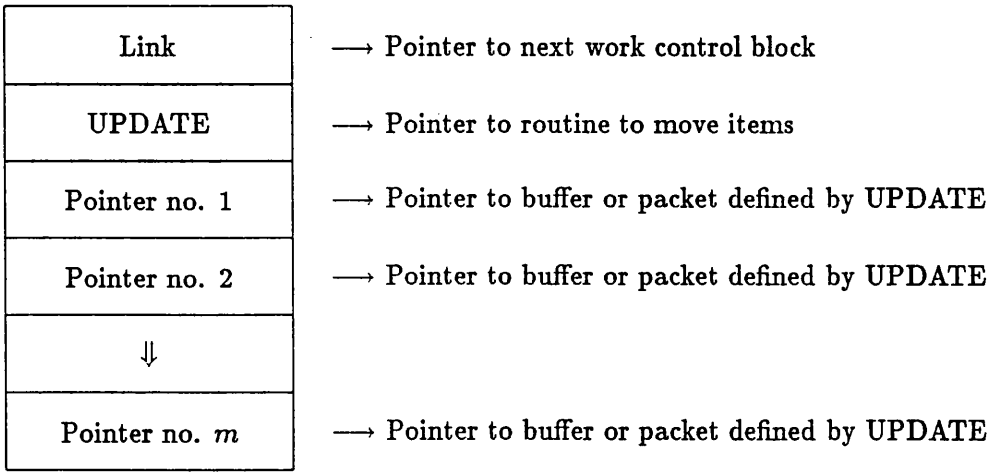


Figure 7.6: Work control block data structure

the instructing packet from the control task, the necessary computations are carried out, and when complete, the packet is returned to the control task.

The third function is carried out at the request of the monitor program when the QUIT command has been entered and the various tasks are required to close themselves down.

7.4.5 Estimator Task

The estimator task is essentially similar to the model task, but has a total of four functions;-

- compute next estimates of states and outputs
- update estimates with new output measurements
- compute the lower half of the feedback vector
- self-destruct

The first and third functions of the estimator task are essentially similar to the first and second functions of the model task. However, when the model task has computed the new outputs, these are used to update the

estimates produced by the estimator, and this is the purpose of the second function.

The self-destruct function of this task is identical to the self-destruct function of the model task, and is only executed at the request of the monitor task when the QUIT command has been entered.

7.5 Graphics Software

The data files produced by the simulations containing time histories can be plotted and graphically manipulated by an off-line plotting program. The structure of the data files has already been discussed in Section 7.4.1, and this structure is used by the plotting program for a variety of purposes.

The results plotting program can be invoked by entering the command "res-plot" with an appropriate argument string. The arguments can be used to direct the graphical output to be displayed on the colour monitor via the EFCIS card, or the terminal screen via the Tektronix driver. In addition, a file can be specified which contains a series of commands for the program, avoiding the need for user interaction, so that it can be run as a background task. The following description of the program's operation is based on its interactive mode, although the only difference in operation in the command file mode is that the program reads commands from the specified file.

When invoked, the results plotting program prompts the user for the name of the file where the results are stored, which it then reads, identifying the parameters of the simulation from which the data was produced, and noting the variables that were logged.

When this is complete, the program enters an interactive procedure, issuing a prompt and waiting for a command. In a similar manner to that of the monitor program of the simulation, when a command is entered at the terminal, the program attempts to identify it, and then executes the appropriate function, returning a prompt when finished. The following commands are available;-

- HELP
- CLEAR
- WINDOW
- STYLE
- PLOT

- FIGURE TITLE

The HELP command displays a list of the available commands with a short description of their functions.

The CLEAR command is used to reset the display area to the background colour, effectively clearing the screen.

The WINDOW command is used to define an sub-area of the display area on which subsequent plots will be displayed. This is used to format several plots on the display area, either for easy comparison, or for convenient hard copies. The following window areas are available;-

WHOLE	LARGE	A4L	A4R
UL	UR	LL	LR
A4UL	A4LL	A4UM	A4LM
A4UR	A4LR		

The WHOLE area, as its name implies, causes the whole of the display area to be utilised, and this is the default window. The LARGE area is slightly smaller than WHOLE, and allows space for a title string to be added to the display. The A4L and A4R are complementary windows for positioning two plots on either half (left or right, respectively) of an A4 area. The UL, UR, LL, LR windows divide the display area into a set of four quadrants (upper left, upper right, lower left, lower right, respectively). The A4UL, A4LL, A4UM, A4LM, A4UR, and A4LR windows can be used to position six plots on an A4 area (upper left, lower left, upper middle, lower middle, upper right, lower right, respectively).

The STYLE command can be used to alter the style of the plots produced, with facilities such as colour changing, and plot grid on or off.

The PLOT command is used to produce the graphs, and expects the variable type and number that is to be plotted as part of the argument string. The routine checks that the requested variable has been recorded in the data file, returning an appropriate message if it is not able to find the variable, specifying the reason for failure. Other arguments can be used to specify a particular portion of the time history rather than the whole history, and to request the option of outputting data for subsequent transfer to the mainframe machine to obtain high quality hard copies on a barrel plotter. If the hard copy request argument is specified, when the time history has been plotted the user is asked whether a hard copy of the plot is required, or not. If the response is affirmative, the co-ordinates of the plot are written to a file created automatically by the program, preceeded by the number of co-ordinate pairs in the plot, and followed by the X axis and Y axis title

strings. When the plotting program is terminated, if a hard copy data file has been created, a zero is added to the end of the file and then the file is closed. The data structure created is subsequently used by the plotting program on the mainframe to control the formatting of plots automatically (see Section 7.6.1). A diagrammatical representation of this data structure is shown in figure 7.7.

It should be noted that the data files created for transfer to the mainframe are ASCII files, not binary files, for two reasons. Firstly, the Kermit file transfer program is more reliable when transferring ASCII files, as opposed to binary files, and secondly, the representation of floating point numbers under the VAX/VMS operating system is not of the same format as that used by the TRIPOS operating system.

The FIGURE TITLE command is used to add a title string to the display area, which is particularly useful for identifying hard copies obtained via the screen dump facility in the Tektronix emulation firmware incorporated in the terminal.

7.6 Mainframe Support

The data link with the mainframe machines has already been mentioned. At the experimental rig's current location, a group of mainframe machines are available via a multiplexing data link switch, all of these being DEC machines, running the VAX/VMS operating system. In particular, a pair of VAX 11/785 machines, a VAX 11/750 machine, and a VAX 11/730 machine are available, and all of these machines are interconnected by a high speed data transfer network.

The mainframe machines are used for two purposes;-

- high quality graph plotting
- control systems design

The justification for the first use is the access to a Benson barrel plotter which can produce high quality plots with a resolution of 0.05mm. The justification for the second use is the availability of proven reliable numerical computation routines, such as eigenanalysis routines, which are essential tools in linear multivariable control systems design. The software which has been developed for these purposes is now described.

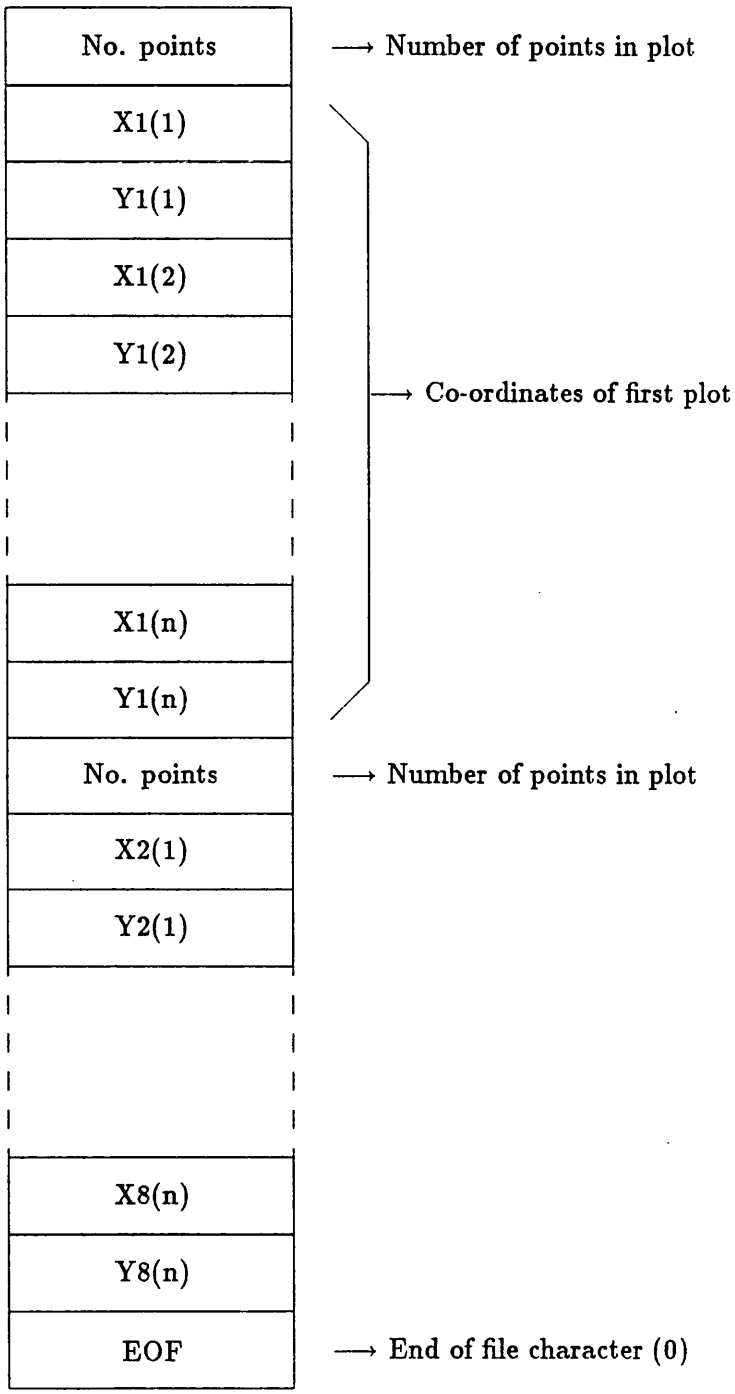


Figure 7.7: ASCII character plot file data structure

7.6.1 High Quality Graph Plotting

The creation of the data files in ASCII character representation has already been described in Section 7.5. These files are transferred to any of the mainframe machines via an RS232 serial data link using the Kermit programs (see Section 7.3.1). The program M68KPLOT can then be invoked to produce data files suitable for plotting on the Benson plotter (or alternatively, a Gould plotter, although the resolution of the Gould plotter is much lower than that of the Benson plotter). The structure of the data files allows the program to ascertain the number of graphs in the file, the number of points in each graph, and the X axis and the Y axis titles for each graph automatically. The maximum permissible number of graphs in a data file is set at eight, because more than this produces graphs which are too small to be useful when all of them are plotted on an A4 size area. The program then offers the user a full screen size inspection of each graph, then offers to format the graphs.

The graphs are plotted using routines from the GINO-F [109] and the GINOGRAF [110] graphics libraries, and these are used to generate a temporary "pseudo-code" file, which is device independent. The picture elements in this file (which correspond to the individual graphs) are then extracted, scaled, and postioned automatically onto an A4 size plotting area, with either horizontal or vertical orientation of the individual graphs selected by the user. A box approximately 2mm over A4 size is drawn around the plots, and a title string for the complete plot can be added if required.

The interactive nature of the program means that the user can elect to alter certain aspects of the overall plot, such as the position of the title string, or the horizontal/vertical orientation of the graphs.

When the user is satisfied with the overall plot, the required plotter driver is invoked, thus producing a data file suitable for plotting. On termination of the program, the resulting data file can be queued to the appropriate plotter for plotting.

In a similar manner to the results plotting programs developed for the experimental rig computing system, this plotting program can also be run as a background task, reading its commands from a file.

When the completed plot is obtained, the paper can be trimmed to just inside the plotted box to produce an A4 sized page ready for insertion into a document.

The GINO-F and GINOGRAF routines are written in FORTRAN, and are designed to be called from FORTRAN programs, thus FORTRAN was

used to code the plotting program.

7.6.2 Control Systems Design Routines

It has already been mentioned that the primary reason for developing the control systems design software on the mainframe machines is the availability of proven reliable numerical computation software, in particular, the NAG library [111]. The development and testing of reliable numerical routines is a specialist field in its own right, and besides being a non-trivial problem, is exceedingly time-consuming and outside the scope of this work. Thus it was decided to make use of the routines in the NAG library for such functions as eigenanalysis, singular value decomposition, and matrix inversion (see Appendix B).

The version of the NAG library that is available is written in FORTRAN, so FORTRAN has been used to code the control systems design programs. As all the design algorithms are based on the state space representation of dynamic systems, a variety of matrix operations are frequently used. Thus a library of subroutines was developed which forms the basis of the main programs, and these subroutines and programs are now described.

7.6.2.1 Subroutine Library

The following subroutines form the basis of the control systems design programs;-

EIGVAL	EIGVALVEC	FIXD-MODES	LYAPUNOV
MATADD	MATINV	MATMUL	MATRAN
MATRANK	MATSUB	NON-DEGEN	PSEINV
RICATTI	TXZEROS		

The subroutine EIGVAL requires a real square matrix and returns the eigenvalues of the matrix in real and imaginary components. The details of the algorithm can be found in Appendix B.1.

The subroutine EIGVALVEC is essentially similar to EIGVAL except that in addition to the eigenvalues, the eigenvectors are also returned, again in real and imaginary components.

The subroutine FIXD-MODES computes the fixed modes (or approximately fixed modes) of a system given the state space matrices describing the system. The algorithm used for this subroutine is based on the algorithm given in [112].

The subroutine LYAPUNOV requires matrices A and Q, and computes

the solution P to the Lyapunov type matrix equation $A^TP + PA = -Q$, using the algorithm described in Appendix B.4.

The subroutine MATADD requires two real matrices, which it adds together and returns the result.

The subroutine MATINV requires a real square matrix, which it checks for singularity, then computes the inverse, returning the result. Details of the algorithm used can be found in Appendix B.2.

The subroutine MATMUL requires two real matrices, computes the product, and returns the result.

The subroutine MATRAN requires a real matrix, computes the transpose, and returns the result.

The subroutine MATRANK requires a real matrix, and computes the singular values of the matrix. The number of non-zero singular values indicates the rank of the matrix. The algorithm used to compute the singular values is described in Appendix B.2. This method of computing the rank of a matrix is the most reliable method known to date.

The subroutine MATSUB requires two real matrices, computes the difference matrix, and returns the result.

The subroutine NON-DEGEN attempts to ascertain whether or not a system is non-degenerate. The algorithm used in this subroutine is based on the algorithm given in [112].

The subroutine PSEINV requires a real matrix, and computes the generalised inverse (pseudo-inverse) via the singular value decomposition of the matrix. Details of the algorithm used are given in Appendix B.2.

The subroutine RICATTI requires real matrices A, B, Q, and R, and computes the matrix P, where $0 = A^TP + PA + Q - PBR^{-1}B^TP$, and also the matrix F, where $F = -R^{-1}B^TP$. Details of the algorithm used are given in Appendix B.3.

The subroutine TXZEROS computes the transmission zeros of a system, given the state space matrices. The algorithm used is based on the algorithm given in [112].

7.6.2.2 Main Programs

A suite of programs has been developed for implementing a variety of system order reduction algorithms and feedback law design algorithms. The various algorithms have already been discussed in Chapters 3 and 5, and the programs are now listed and briefly described as follows;-

QCOVER	MCA	IMSC	POLPLC
PPOBDES	LUEN	RMDES	SERVO
LQREG	LQTRK	WIENER	LQGDES
HACLAC	LACHAC		

The program QCOVER is an implementation of the "Q-cover" controller reduction algorithm (which is described in Chapter 3), which attempts to produce reduced order controllers with Markov parameters and output covariances which match those of the full order controller. The program starts off by computing a first order controller, then a second order controller, and so on, incrementing the controller order by one each time until the full order controller is obtained, evaluating the closed loop poles of the system with each reduced order controller. However, in almost all cases (see Chapter 8 for the only exception), the program prematurely terminated due to numerical problems, such as floating point overflows, before a reduced order controller was obtained which resulted in a stable closed loop system.

The program MCA is an implementation of the Modal Cost Analysis model reduction algorithm (which is described in Chapter 3), which computes a "cost" for each mode in the given model. The modes are then ordered in terms of cost from highest to lowest. An option is then available to the user to produce a reduced order model by entering the numbers of the modes to be retained in the reduced order model.

The program IMSC is an implementation of the Independent Modal Space Control technique (see Chapter 5 for details). An output feedback matrix is computed by first computing input and output decoupling matrices, and then computing a feedback matrix by assuming that the system is decoupled. The final feedback matrix is obtained as the product of the input decoupling matrix, the decoupled feedback matrix, and the output decoupling matrix. When the final feedback matrix has been computed, the closed loop poles of the system are computed to ascertain the effectiveness of the decoupling process.

The POLPLC program is an implementation of the pole placement design algorithm based on the transformation of the system to controllable canonical form. Whilst this program produces good results when the order of the problem is reasonably small, and the dynamics of the problem are not too numerically stiff, high order problems and numerically stiff problems suffer from numerical difficulties such as inaccurate results, or overflow problems.

The PPOBDES program is an implementation of a full order asymptotic state estimator design method using the pole placement algorithm based

on the transformation of the system to observable canonical form. This algorithm is essentially similar to the algorithm in the POLPLC program, and as such it suffers from the same numerical difficulties.

The LUENOBS program is similar to the PPOBDES program, except that it is a minimal order state estimator design method. It is also based on the transformation of the system to observable canonical form, and consequently suffers similar problems as the POLPLC and PPOBDES programs.

The RMDES program is an implementation of an output feedback design method which is based on a model which is comprised of the rigid modes only. Thus the feedback matrix is computed with no knowledge of the elastic modes, but is constrained to be diagonal with the diagonal elements being greater than zero. This is based on the stability properties discussed in Chapter 4.

The SERVO program is a partial implementation of the Robust Multivariable Servomechanism design method (see Chapter 5). At this time, the tests for ascertaining the existance of a solution have been implemented, but a method for automatic design of the compensators has not yet been implemented.

The program LQREG obtains a solution to the Linear Quadratic Regulator problem. That is, the solution to the associated algebraic matrix Riccati equation is computed, and subsequently, the optimal state feedback matrix is computed.

The program LQTRK is essentially similar to the program LQREG, but it solves the "tracking" problem rather than just the "regulator" problem. Besides computing the optimal state feedback matrix, a dynamic input compensator is also computed which will cause the system outputs to track the inputs to the compensator.

The program WIENER is the dual of the program LQREG, as it computes the solution to the optimal dynamic state estimator problem, that is the Linear Quadratic Estimator problem. Again, the solution to the associated algebraic matrix Riccati equation is computed, and subsequently the estimator gain matrix is computed.

The program LQGDES is essentially a combination of the LQREG and WIENER programs, such that the optimal state feedback matrix and the optimal state estimator gain matrix are computed for the given model. When both stages are complete, an option is available to the user to compute the closed loop poles of another model subject to the designed feedback system, which can be used to evaluate a low order design on a full order model, or a nominal design on a perturbed model.

The program HACLAC is basically an extension to the LQGDES program which carries out the same functions as the LQGDES program, but includes the option of adding rate output feedback terms for additional damping control, thus combining a state estimator/state feedback controller with an output feedback controller.

The program LACHAC is essentially similar to the program HACLAC, the only difference is that the rate output is added to the plant before the design of the state estimator/state feedback controller.

7.7 Summary

In this chapter, the software developed to provide support for the experimental rig, for both the experimental rig computing system and for DEC VAX 11/series mainframe machines has been described. Details of the TRIPOS operating system for the experimental rig computing system have been given, together with a description of the inter-task communication mechanism, as this is utilised in the simulation software.

The Kermit file transfer programs have been introduced, as they are an essential part of the support software, facilitating the transfer of files between the experimental rig computing system and the mainframe machines. The Graphical Kernel System (GKS) has also been introduced, as this graphics system is used for off-line graphical processing on the experimental rig.

The simulation software has been discussed with a model/state estimator type of simulation taken as an example. The task structure has been described, together with details of the function and operation of the individual tasks. The various commands and functions that the simulation monitor program offers for operating the simulations have been detailed. The reasons for designing the simulations in this particular format have also been discussed. The structure for the binary simulation results files has been described to show how the subsequent processing programs can know what to expect in the file after reading just a small block (about 40 bytes).

Software for graphical processing of simulation results on the experimental rig computing system using GKS routines has been described, together with details of the creation of ASCII character files for subsequent transfer to a mainframe machine to obtain high quality plots. The structure of these data files has also been described to show how the graph plotting program on the mainframe can be controlled automatically by the data in the file itself. The mainframe graph plotting software utilising the GINO-F and

GINOGRAF libraries has subsequently been described.

An essential use of the mainframe machines is for control systems design. Details of the subroutine library which has been developed to form the basis of the main programs have been given, with a brief description of the function of each routine. Subsequently, details of the programs used for system reduction problems and feedback law design problems have been given, together with a brief description of their function. The use of these programs for application to a model of the experimental beam is discussed in the next chapter.

Chapter 8

Controller Design for Experimental Beam

8.1 Introduction

In this chapter the various techniques that have been discussed in Chapters 3 and 5 are applied to the problem of designing a controller for the experimental beam, the details of which have already been given in Chapter 6.

It is assumed that a torque actuator is available which acts about the pivotal axis, and that measurements of position and rate of the beam about this axis are available. The controller is required to provide signals for the actuator such that the position of the beam about the pivotal axis is controlled. This configuration was chosen because of the obvious parallel to the single-axis attitude control problem of a spacecraft.

In the first section, the development of mathematical models for the beam is described and numerical details are given for the nominal and perturbed models.

Subsequent sections describe the application of the techniques discussed in earlier chapters for designing feedback laws for flexible structures to the experimental beam, giving numerical results and simulation results. It was intended to implement the successful designs as real-time controllers so that complete verification of the designs could be carried out. However, as was explained in Chapter 6, the computational restrictions meant that such implementation was not possible, and thus results of such exercises cannot be presented for comparison with the simulation results.

The penultimate section of this chapter draws comparisons, and includes

comments regarding the application of the various techniques to the experimental beam, and suggests which techniques would be suitable for application to more realistic and complex problems.

8.2 Development of Mathematical Models of the Experimental Beam

The basic configuration of the experimental beam has already been described in Chapter 6. Because it was desired to be able to model the displacements of the beam at various points along its length, the finite element technique described in Chapter 2 was used to represent the structure. In particular, the finite element structural analysis program NASTRAN [32], [33] was utilised to obtain modal frequency and modeshape data. The matrices defining the corresponding state space models were formulated in the manner described in Chapter 2.

It has already been mentioned in Chapter 6 that holes were cut in the beam at various points along its length to facilitate the mounting of additional masses which could be used to alter the dynamic characteristics of the beam. The removal of this material obviously reduces the stiffness and the mass of the beam, and thus these holes were included in the finite element representation.

Each 10cm length of the beam (the distance between each hole) was represented by a pair of beam elements, one being essentially the reflected image of the other. The mountings for the two halves of the beam to the pivot point were also represented by beam elements. Thus figure 8.1 shows the representation of the beam using three types of beam elements, where the element type representing the beam mountings is denoted as A, and the two elements representing a section of the beam are denoted as B and C, and the numbers represent the finite element nodes. The masses used to alter the dynamic properties of the beam were represented by lumped masses at the appropriate node. The various properties of the elements are shown in table 8.1.

The baseline model, referred to as the nominal model, is comprised of the beam plus two masses, one at node 2 and one at node 42 (see figure 8.1), that is at the two ends of the beam. This configuration was analysised using the NASTRAN program to obtain the mode frequencies and the modeshapes. The modal frequencies up to 100Hz are shown in table 8.2, together with the slope of the modeshapes at the pivotal axis. Because the modeshapes

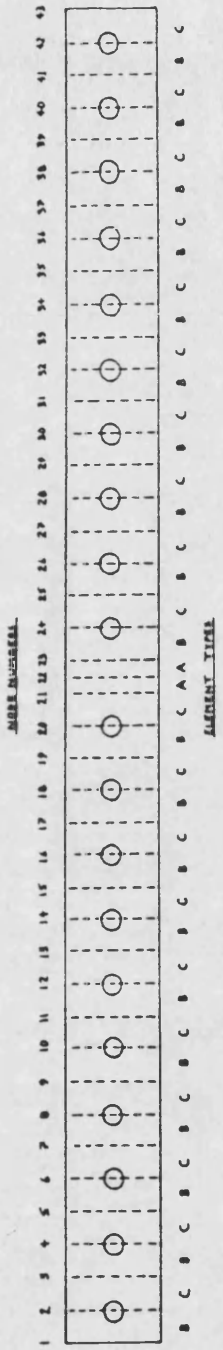


Figure 8.1: Representation of experimental beam using finite elements

Element type	L (m)	A (m^2)	I1 (m^4)	I2 (m^4)	J (m^4)	E ($N.m^{-2}$)	Density ($Kg.m^{-3}$)
A	0.075	4.50×10^{-4}	3.53×10^{-8}	3.98×10^{-7}	1.41×10^{-7}	6.89×10^{10}	2.72×10^{3}
	0.05					6.89×10^{10}	2.72×10^{3}
B	0.00L	4.50×10^{-4}	3.38×10^{-10}	8.44×10^{-7}	1.35×10^{-9}		
	0.60L	4.50×10^{-4}	3.38×10^{-10}	8.44×10^{-7}	1.35×10^{-9}		
	0.68L	3.96×10^{-4}	2.97×10^{-10}	8.42×10^{-7}	1.19×10^{-9}		
	0.78L	3.60×10^{-4}	2.70×10^{-10}	8.37×10^{-7}	1.08×10^{-9}		
	1.00L	3.42×10^{-4}	2.57×10^{-10}	8.32×10^{-7}	1.03×10^{-9}		
	0.05					6.89×10^{10}	2.72×10^{3}
C	0.00L	3.42×10^{-4}	2.57×10^{-10}	8.32×10^{-7}	1.03×10^{-9}		
	0.22L	3.60×10^{-4}	2.70×10^{-10}	8.37×10^{-7}	1.08×10^{-9}		
	0.32L	3.96×10^{-4}	2.97×10^{-10}	8.42×10^{-7}	1.19×10^{-9}		
	0.40L	4.50×10^{-4}	3.38×10^{-10}	8.44×10^{-7}	1.35×10^{-9}		
	1.00L	4.50×10^{-4}	3.38×10^{-10}	8.44×10^{-7}	1.35×10^{-9}		
Loads	0.02	4.54×10^{-3}	3.15×10^{-6}	3.15×10^{-6}	3.65×10^{-6}		7.85×10^{3}

Table 8.1: Properties of elements

corresponding to the symmetric modes have no displacement at the pivotal axis (the mid-point of the beam), these modes will obviously be both uncontrollable and unobservable, and thus can be safely ignored, as any feedback law for this configuration cannot affect them, or be affected by them. Thus, the resulting nominal model is 10th order, with one input and two outputs.

Frequency in Hertz	Slope of modeshape at node 22
0.0	-0.6080
1.3083	0.0
6.7022	-2.5778
11.6665	0.0
22.6640	4.1875
31.5122	0.0
43.7555	-5.0041
57.2307	0.0
74.1841	6.3486
98.1012	0.0

Table 8.2: Modal data for nominal beam model

In an attempt to investigate the sensitivity of controller designs to model parameter variations, two other models were derived which correspond to slightly different configurations with different dynamic properties, and these models are referred to as perturbed model no. 1 and perturbed model no. 2. The configuration for the perturbed model no. 1 is exactly the same as the nominal model but with additional masses of 0.8Kg at nodes 6, 10, 14, 18, 26, 30, 34, and 38. Thus the overall mass has been increased, but the mass distribution is about the same as the nominal model. Thus as would be expected, the modal frequencies are lower than those of the nominal model, but the modeshapes are essentially the same shape as those of the nominal model.

The details (below 100Hz) are shown in table 8.3, where it can be seen that the symmetric modes again produce no displacements at the mid-point of the beam, and so can be ignored, hence the resulting model is 12th order.

The configuration for the perturbed model no. 2 is exactly the same as

Frequency in Hertz	Slope of modeshape at node 22
0.0	-0.4526
1.0699	0.0
3.9524	1.3357
6.6306	0.0
12.7109	2.2063
18.0398	0.0
26.0370	2.6777
34.0486	0.0
43.4689	2.6892
53.7234	0.0
65.9434	1.0244
67.0419	0.0

Table 8.3: Modal data for perturbed model no. 1

the nominal model, but with additional masses of 0.8Kg at nodes 4, 6, 8, 10, 24, 26, 28, and 30. Thus the overall mass has been increased from the nominal configuration, and the mass distribution has been altered. This latter point is particularly interesting because the point of symmetry of the symmetric modes is no longer at the mid-point of the beam, and thus, as can be seen from table 8.4, the modes that could be safely ignored for the nominal model and the perturbed model no. 1 cannot now be safely ignored. Hence the resulting model is 26th order.

It should be noted that these models do not incorporate any actuator or sensor dynamics, and this as largely because a set of suitably defined dynamics does not exist at present. However, the sensors used for the beam do have negligible dynamics (at least up to 100Hz), and so their neglection is not too severe an assumption. Also, dynamic equations for the actuator are difficult to define until a suitable working model has been developed.

Frequency in Hertz	Slope of modeshape at node 22
0.0	-0.4455
1.1035	-0.1770
4.0851	-1.4406
6.9374	-0.2794
15.8492	2.6826
22.2640	0.4155
28.4962	3.0400
41.1839	-0.4851
50.7463	3.1042
61.2881	-0.3719
72.4321	-4.0637
96.2815	0.0559
100.9000	3.7406

Table 8.4: Modal data for perturbed model no. 2

8.3 State Feedback Based Controllers

The controllers considered in this section are based on state feedback, and utilise a dynamic state estimator of some form to produce estimates of the states required for feedback. Following the techniques discussed in earlier chapters, two approaches will be examined. The first approach is to reduce the order of the model, and then design a controller based on the reduced order model, and the second approach is to design a controller based on the full order model, and then reduce the order of the controller.

8.3.1 Model Order Reduction — Controller Design

The first approach used follows the model order reduction — reduced order controller design philosophy. A modal cost analysis of the nominal model was carried out using the MCA program, and the resulting costs for each mode are shown in table 8.5. A graph of model error against the number of modes retained in a reduced order model (where the modes have been ordered in terms of cost, highest to lowest) is shown in figure 8.2.

These results show how the modal costs decrease as the frequency in-

Mode no.	Modal Cost
2	22.375
3	17.415
4	12.882
5	12.232

Table 8.5: Model cost analysis for nominal beam model

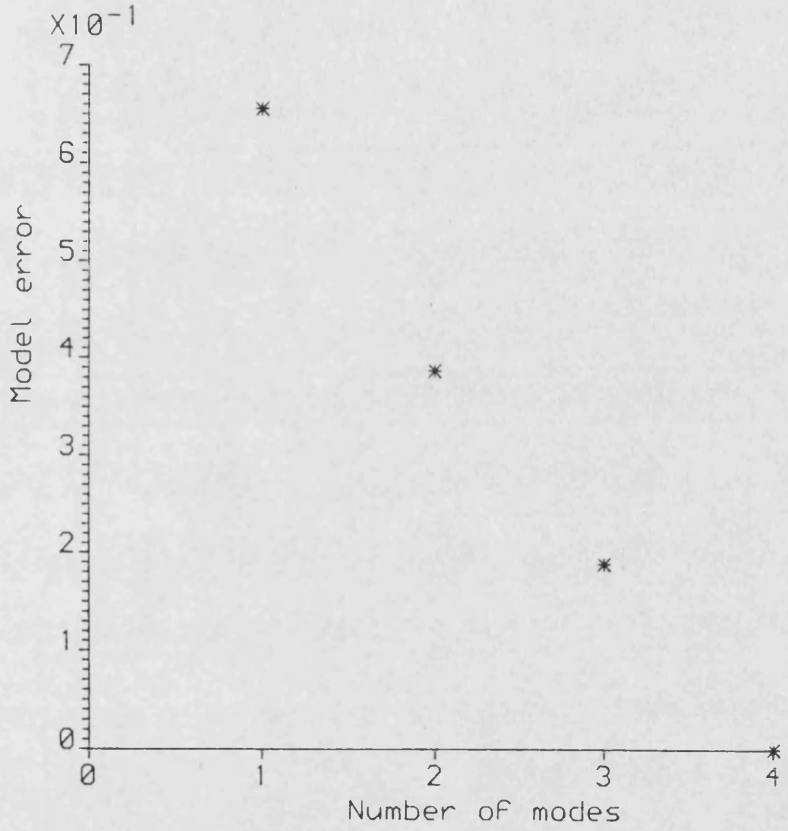


Figure 8.2: Model error against number of modes for nominal beam model

creases, as might be expected for a uniform beam. However, it should be noted that this will not generally be the case for more complex structures as will become apparent in the next chapter.

The desire to control the rigid mode plus the first two elastic modes resulted in a model of 6th order, which has a model error of 38.7%.

The required state feedback matrix was computed using the LQREG program, with a range of weighting matrices. This enabled the effects of the choice of weightings on the position of the closed loop poles to be examined. The results are shown in table 8.6, from which it can be seen that the effects predicted by the theory described in Chapter 5 do indeed occur.

The solution when $Q = R = I$ was chosen as a candidate for further evaluation, and as a basis for comparison with alternative controller designs. Thus the 6th order design model subject to this exact state feedback law (that is assuming that the states are available exactly) acts as a basis for examining the effects of using state estimators to reconstruct the states. The angular position and angular rate responses of the beam to a unit step demand are shown in figure 8.3.

The theory developed in Chapter 4 suggests that when the exact states are available for feedback, then the use of a reduced order model for design purposes will not cause instability of the full order system. Thus, as expected, the angular position and angular rate responses to a unit step position demand for the same state feedback law applied to the 10th order nominal model are stable, but contain lightly damped high frequency components corresponding to the modes that were ignored in the design model (see figure 8.4).

Obviously, the next problem is to design a state estimator to produce estimates of the six states used for feedback. Three types of state estimator were examined, a full order state estimator (where full order implies the same order as that of the design model), a reduced order state estimator (with an order equal to the order of the design model minus the number of outputs of the design model), and a Kalman filter (with order equal to the order of the design model).

8.3.1.1 Full Order State Estimator

The program PPOBDES was used to compute an estimator gain matrix using the 6th order design model as a basis. Note that the observable canonical form of the design model has a dynamics matrix comprising a 1×1 block and a 5×5 block, and thus at least two of the poles have to be real to produce

q/r = 0.01	Gain Matrix	[-0.1 -0.5822 -0.009452 -0.028255 -0.006247 0.014399]
	Eigenvalues	-0.74215 + j 142.4 -0.74215 - j 142.4 -0.24642 + j 42.11 -0.24642 - j 42.11 -0.17699 + j 0.17168 -0.17699 - j 0.17168
q/r = 0.1	Gain Matrix	[-0.31623 -1.0679 -0.11566 -0.19286 0.14844 0.12431]
	Eigenvalues	-0.97228 + j 142.4 -0.97228 - j 142.4 -0.45860 + j 42.108 -0.45860 - j 42.108 -0.32461 + j 0.29477 -0.32461 - j 0.29477
q/r = 1.0	Gain Matrix	[-1.0 -2.0725 -0.82706 -0.85023 2.6205 0.71591]
	Eigenvalues	-2.2111 + j 142.38 -2.2111 - j 142.38 -1.3061 + j 42.091 -1.3061 - j 42.091 -0.62961 + j 0.45998 -0.62961 - j 0.45998
q/r = 10.0	Gain Matrix	[-3.1623 -4.5325 -4.8235 -3.0011 31.694 2.8310]
	Eigenvalues	-6.6458 + j 142.25 -6.6458 - j 142.25 -4.0765 + j 41.913 -4.0765 - j 41.913 -1.3731 + j 0.19282 -1.3761 - j 0.19282
q/r = 100.0	Gain Matrix	[-10.0 -11.683 -30.102 -9.8036 317.08 9.3973]
	Eigenvalues	-20.5378 + j 140.78 -20.5378 - j 140.78 -12.7320 + j 40.099 -12.7320 - j 40.099 -6.0174 -1.0142

Table 8.6: Results of LQREG for nominal model with a range of weighting matrices

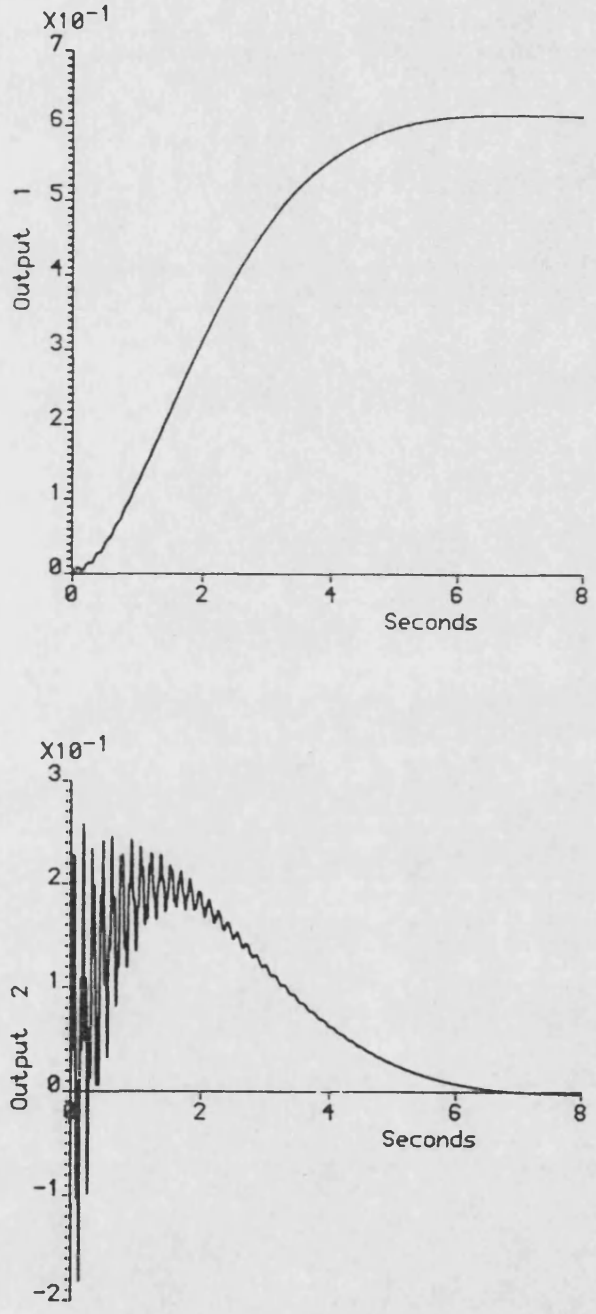


Figure 8.3: Step responses of design model with exact state feedback

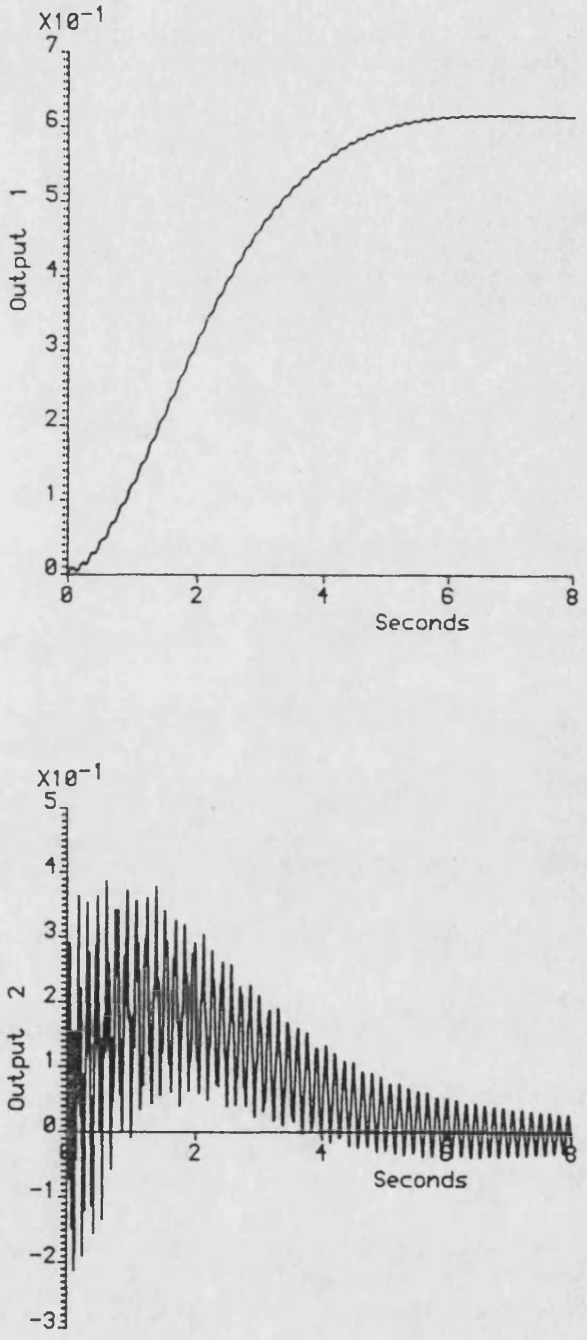


Figure 8.4: Step responses of nominal model with exact state feedback

a real gain matrix. Thus, the desired locations for the estimator poles were selected to be two pairs of complex conjugate poles, and two real poles, at the positions shown in table 8.7. Also shown in the table is the resulting estimator gain matrix.

Gain Matrix	$\begin{bmatrix} -29.092 & -0.08713 \\ 0.0 & -0.59368 \\ 0.0 & -0.04004 \\ 0.0 & 1.20460 \\ 0.0 & 0.20151 \\ 0.0 & 24.1200 \end{bmatrix}$
Eigenvalues	-17.688 -22.041 -17.388 + j 17.759 -17.388 - j 17.759 -21.644 + j 22.020 -21.644 - j 22.020

Table 8.7: Numerical details of full order state estimator

The simulated output responses of the 6th order design model subject to the 6th order state feedback law with the 6th order state estimator are shown in figure 8.5, where it can be seen that no noticeable degradation of the responses occurs when compared to the case of exact state feedback (see figure 8.3).

The simulated step responses of the 10th order nominal model subject to this controller are shown in figure 8.6, where it can be seen that a slight slowing of the responses occurs, but otherwise the response is as designed for.

The sensitivity to parameter variations was simply examined by applying the controller to the perturbed model no. 1 and perturbed model no. 2. The simulated step responses for these cases are shown in figures 8.7 and 8.8, where it can be seen that the performance is essentially unchanged, the only noticeable effect is a slowing of the responses as the parameter variations become more severe.

8.3.1.2 Minimal Order State Estimator

The program LUENOBS was used to compute the various matrices required for a minimal order state estimator. The 6th order design model has two outputs so the resulting estimator is 4th order. As already mentioned, the

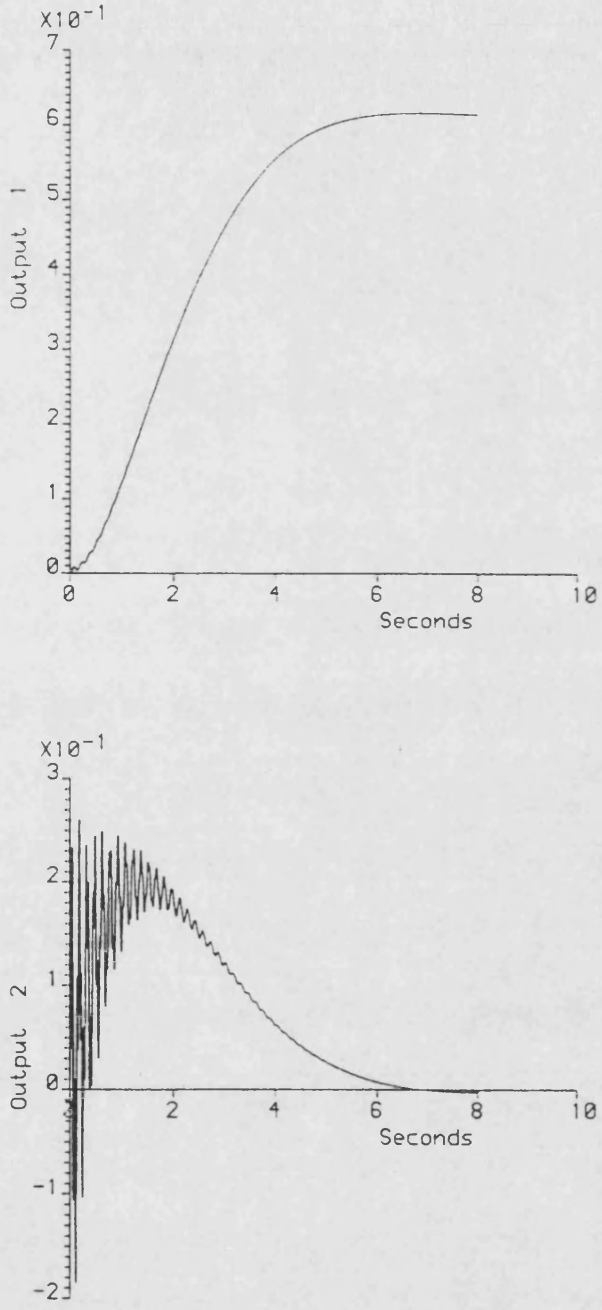


Figure 8.5: Step responses of design model with full order state estimator

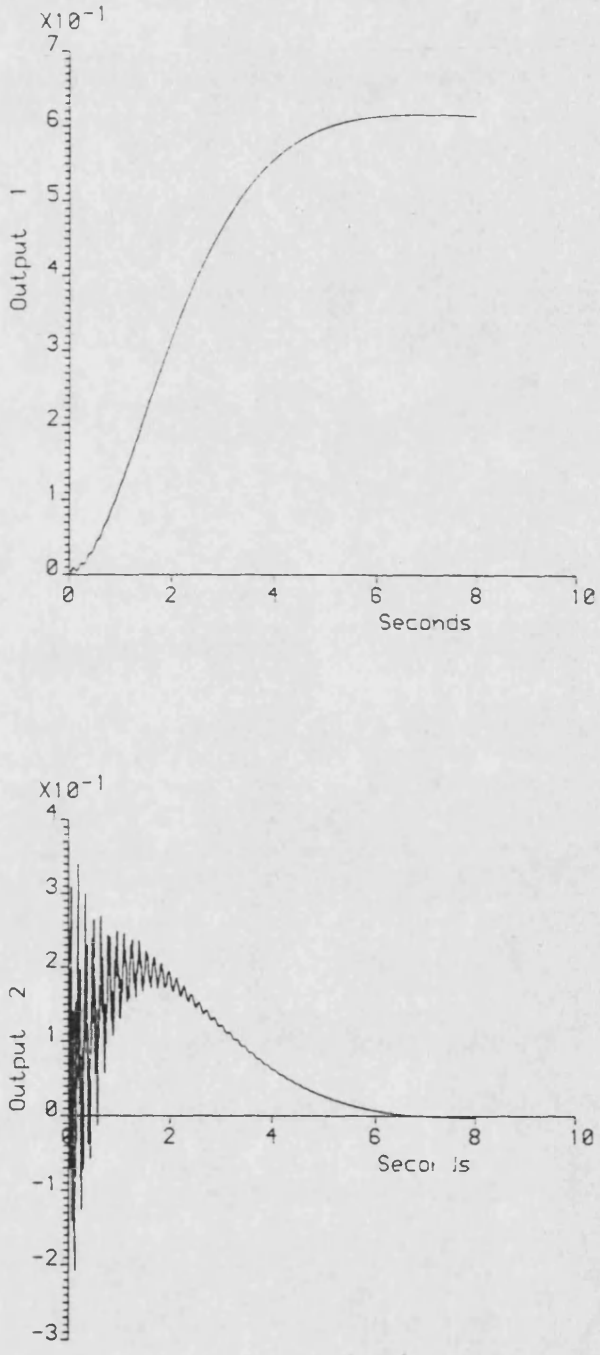


Figure 8.6: Step response of nominal model with full order state estimator

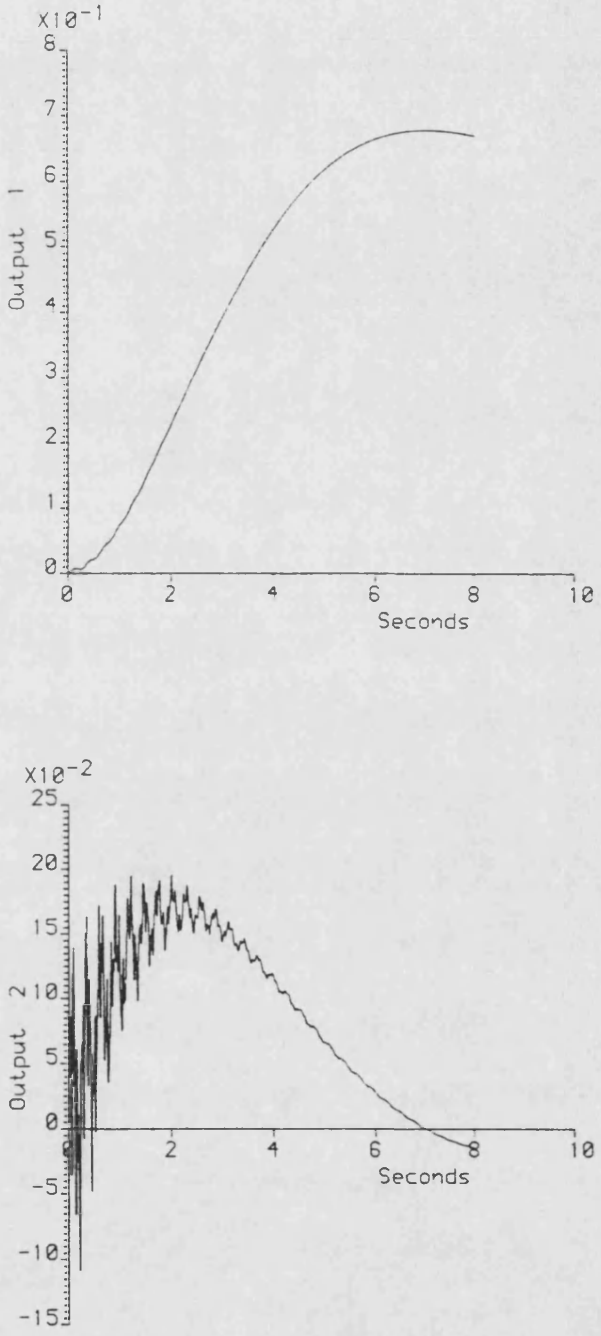


Figure 8.7: Step response of perturbed model no. 1 with full order state estimator

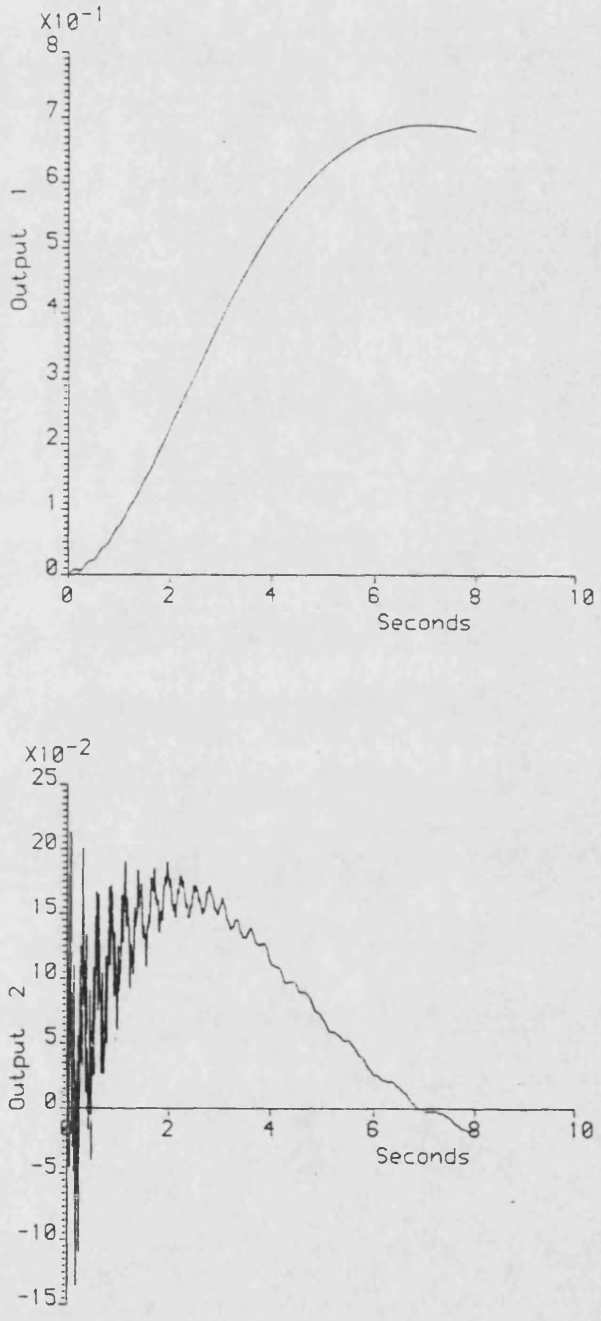


Figure 8.8: Step response of perturbed model no. 2 with full order state estimator

observable canonical form of the design model has a dynamics matrix comprising of a 1×1 block and a 5×5 block, thus the minimal order state estimator dynamics matrix will be composed of one 4×4 block (see Chapter 5). The desired locations for the four estimator poles were chosen to be the same as the two complex conjugate pairs chosen for the full order state estimator, discussed above. This was done in an attempt to match the dynamic characteristics of the two types of estimator as closely as possible to enable better comparisons to be drawn. The poles of the resulting minimal order state estimator are given in table 8.8, together with numerical details of the various matrices.

The simulated step responses of the closed loop system formed by the 6th order design model and the 6th order state feedback matrix with the minimal order state estimator are shown in figure 8.9. Note that there is no noticeable difference between the controller which utilises the full order state estimator and the controller which utilises the minimal order state estimator. This shows that when the model of the system is accurately known, then a minimal order state estimator can perform as well as a full order state estimator.

D Matrix	$\begin{bmatrix} 0.0 & 0.0 & 0.0 & -587200.0 \\ 1.0 & 0.0 & 0.0 & -59770.0 \\ 0.0 & 1.0 & 0.0 & -3072.9 \\ 0.0 & 0.0 & 1.0 & -77.994 \end{bmatrix}$
G Matrix	$\begin{bmatrix} -0.0063028 \\ -0.0082135 \\ 0.0005532 \\ -0.0000106 \end{bmatrix}$
H Matrix	$\begin{bmatrix} 0.0 & -0.25099 \\ 0.0 & -0.22410 \\ 0.0 & -0.00104 \\ 0.0 & -0.00014 \end{bmatrix}$
K1 Matrix	$\begin{bmatrix} -1.6448 & -0.002726 \\ 0.0 & -0.026858 \\ 0.0 & -0.009202 \\ 0.0 & -0.019709 \\ 0.0 & -0.000962 \\ 0.0 & 0.222770 \end{bmatrix}$
K2 Matrix	$\begin{bmatrix} 0.002502 & -8.1485 & -4.57e-16 & -2.48e-13 \\ -8.1485 & -7.42e-18 & -4.05e-15 & 0.0 \\ -0.000613 & 2.1060 & 0.20263 & -3734.8 \\ 2.1060 & 0.20263 & -3734.8 & 1209.3 \\ -1.41e-5 & 0.11337 & 0.12474 & -2299.1 \\ 0.11337 & 0.12474 & -2299.1 & 744.43 \end{bmatrix}$
Eigenvalues	$-17.388 + j\ 17.759$ $-17.388 - j\ 17.759$ $-21.644 + j\ 22.020$ $-21.644 - j\ 22.020$

Table 8.8: Numerical details of minimal order state estimator

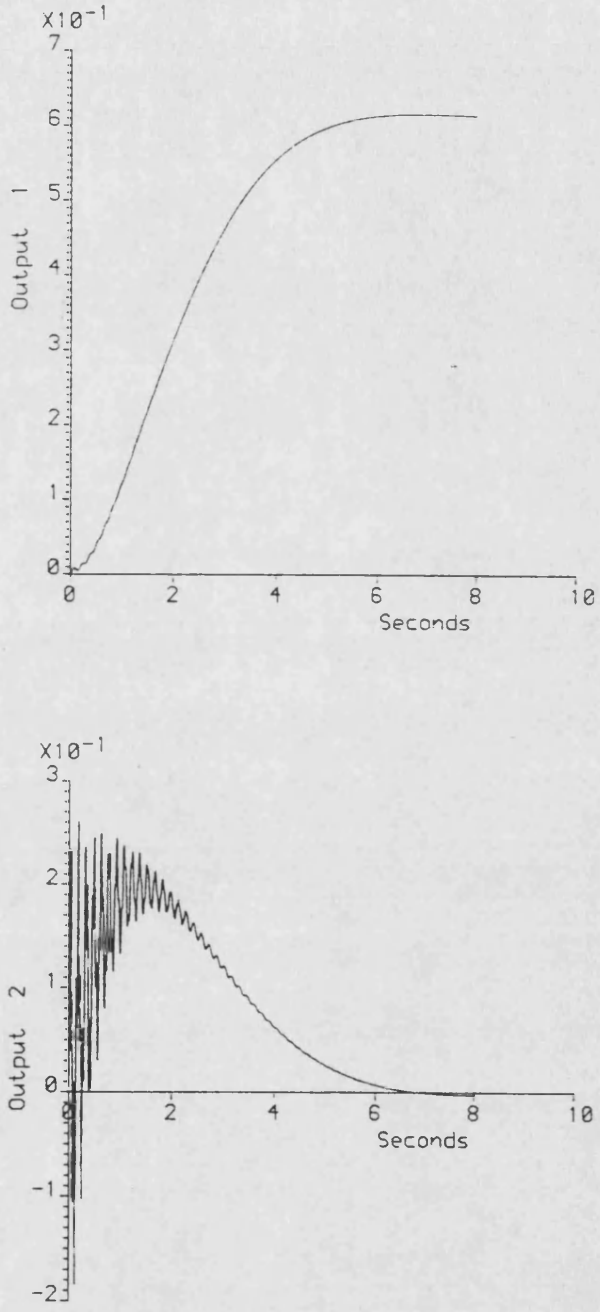


Figure 8.9: Step responses of design model with minimal order state estimator

The step responses of the system formed by the 10th order nominal model and this controller are shown in figure 8.10, where it can be seen again that the responses are essentially the same as those obtained using the full order state estimator (see figure 8.6).

However, it is generally known that minimal order state estimators tend to be more sensitive to parameter variations (or system–model mismatch) than full order state estimators. This is reinforced by the general remarks by Davison and Ferguson [45], where it is shown that generally the more redundancy in a feedback system, then the more robust (that is less sensitive to parameter variations) that system will tend to be. This appears to be confirmed by the simulation of the perturbed models with this controller. The step responses of the perturbed model no. 1, shown in figure 8.11, show the closed loop system to be unstable. As might be expected, the step responses of the perturbed model no. 2, which are shown in figure 8.12, are even worse, the instability being more pronouced.

8.3.1.3 Kalman Filter

The third type of state estimator investigated was a Kalman filter, again based on the 6th order design model. The system and measurement noise covariance matrices were set to unity, and the initial filter gains were chosen at random. The initial filter gains were chosen at random in order to examine the transient behaviour of the filter gains as they migrated towards their steady-state values (assuming a stable system) as defined by the covariance matrices.

The simulated step response of the design model subject to the 6th order state feedback law using the 6th order Kalman filter are shown in figure 8.13, and the twelve filter gain time histories are shown in figures 8.14 and 8.15. Note that the step responses are essentially the same as for the equivalent cases using the full order fixed gain state estimator, or the minimal order fixed gain state estimator. It is also intersting to note that the dynamics of all the filter gains are faster than the overall system responses.

Similarly, the simulated step responses of the 10th order nominal model subject to this controller, shown in figure 8.16 (with the filter gains time histories shown in figures 8.17 and 8.18), are much the same as the comparitive cases with the fixed gain state estimators (see figures 8.6 and 8.10).

The results with the perturbed models is, however, different. In both cases the closed loop systems are stable (see figures 8.19 and 8.22), but the

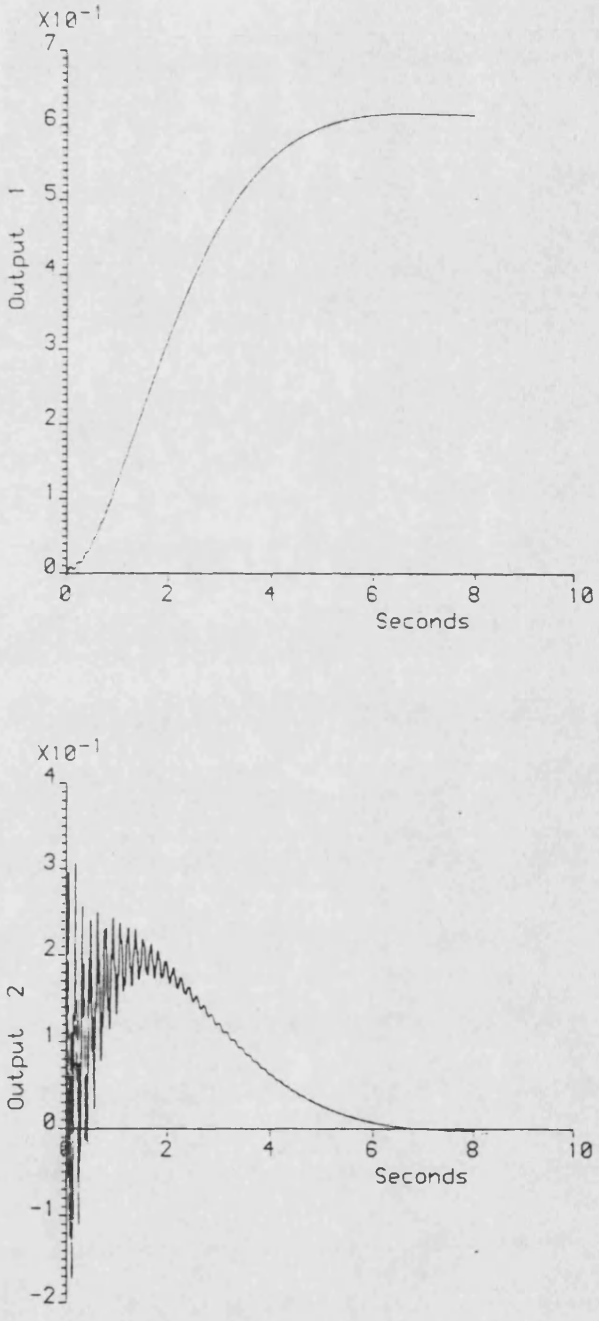


Figure 8.10: Step responses of nominal model with minimal order state estimator

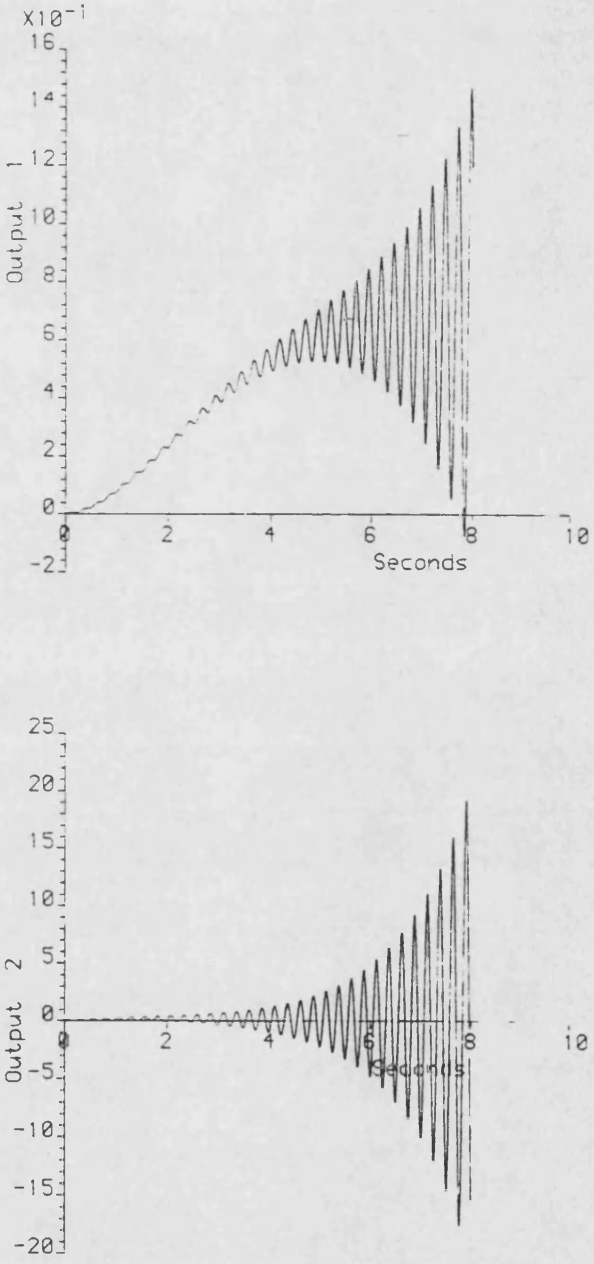


Figure 8.11: Step responses of perturbed model no. 1 with minimal order state estimator

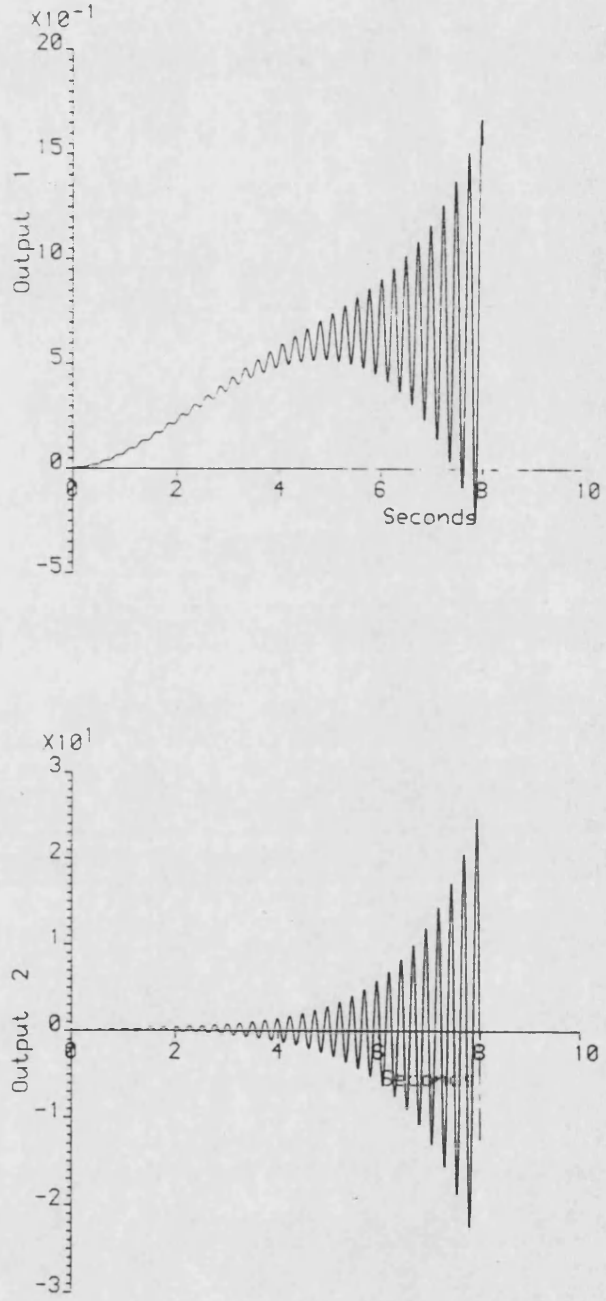


Figure 8.12: Step responses of perturbed model no. 2 with minimal order state estimator

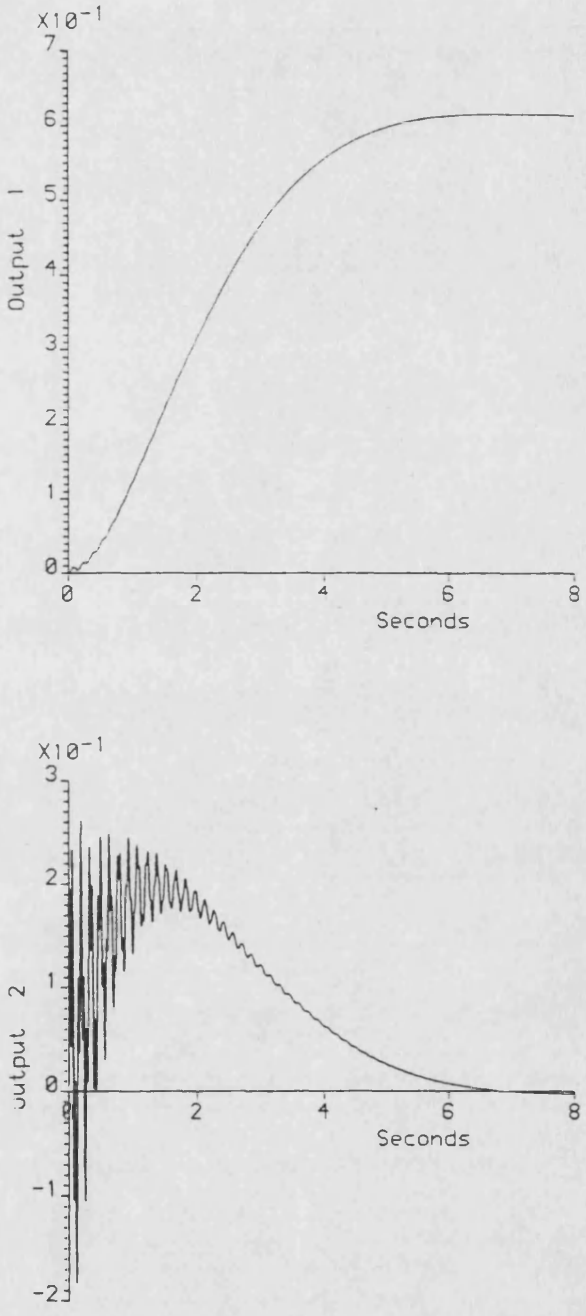


Figure 8.13: Step responses of design model with Kalman filter

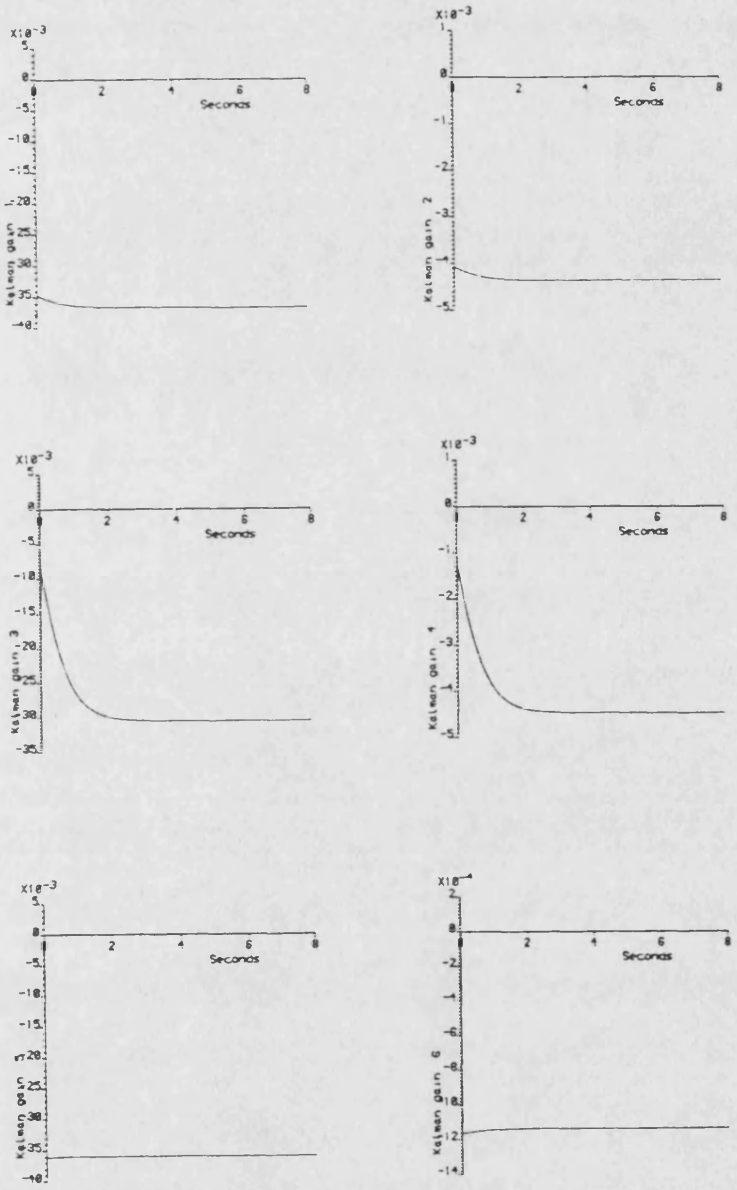


Figure 8.14: Time histories of first six Kalman filter gains for design model

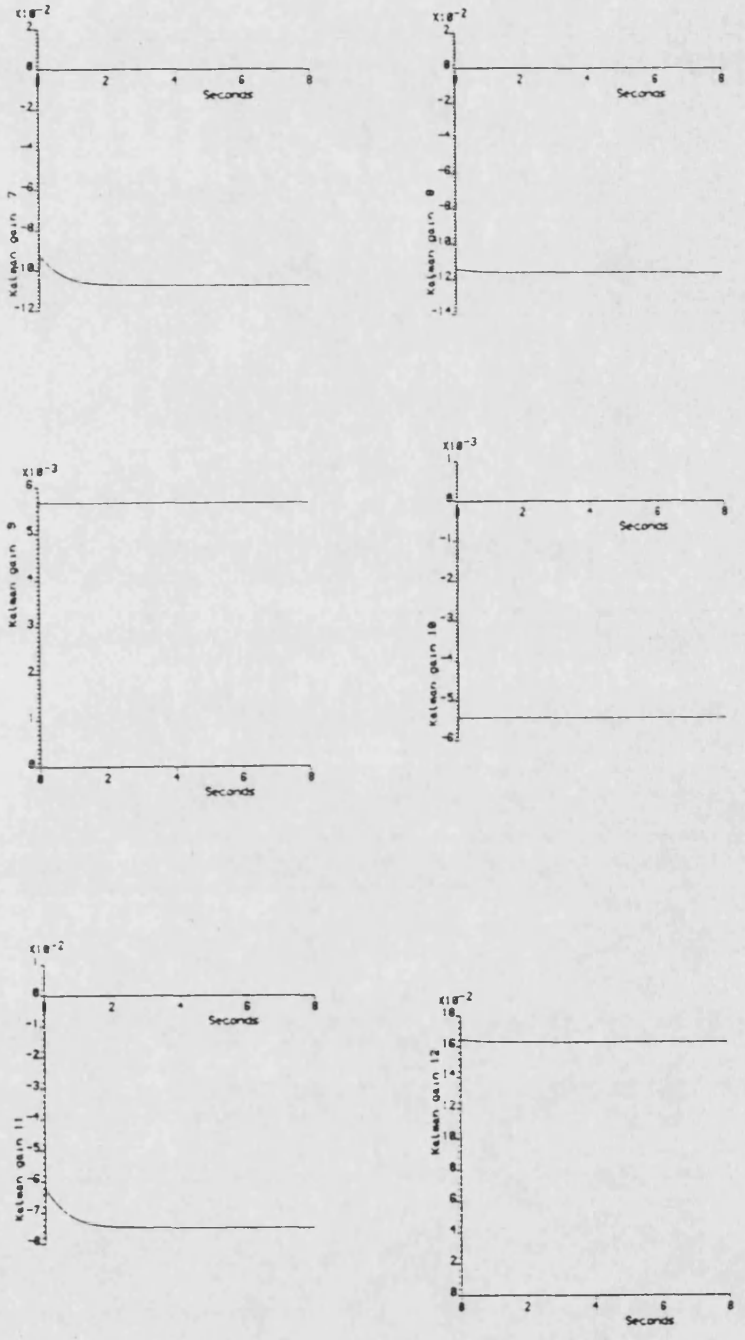


Figure 8.15: Time histories of second six Kalman filter gains for design model

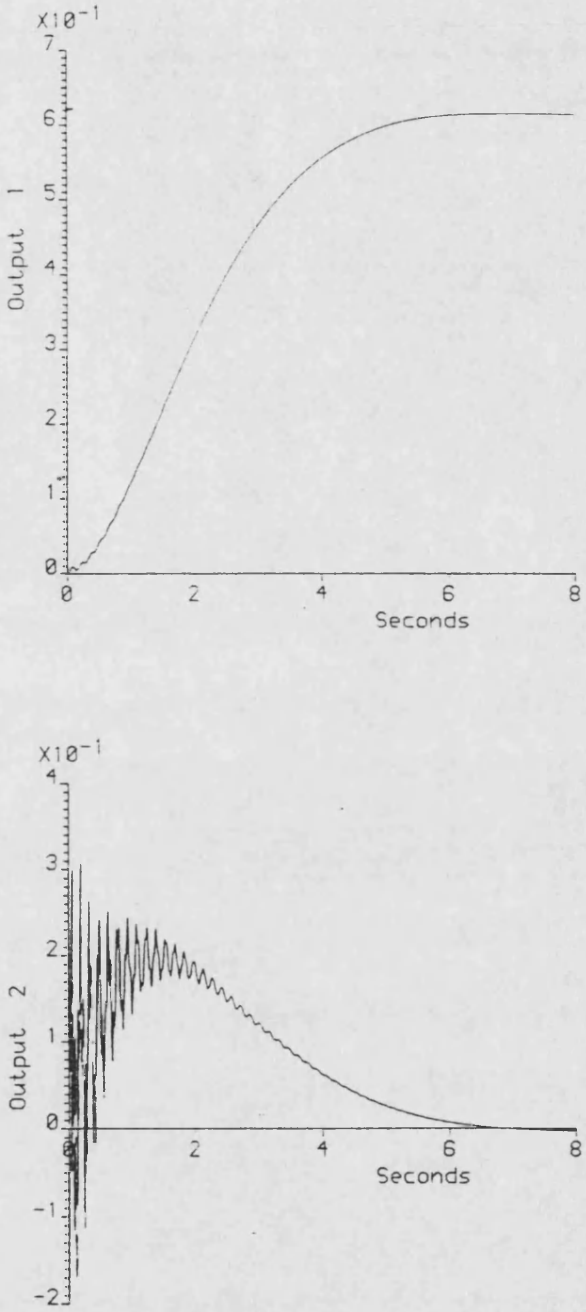


Figure 8.16: Step responses of nominal model with Kalman filter

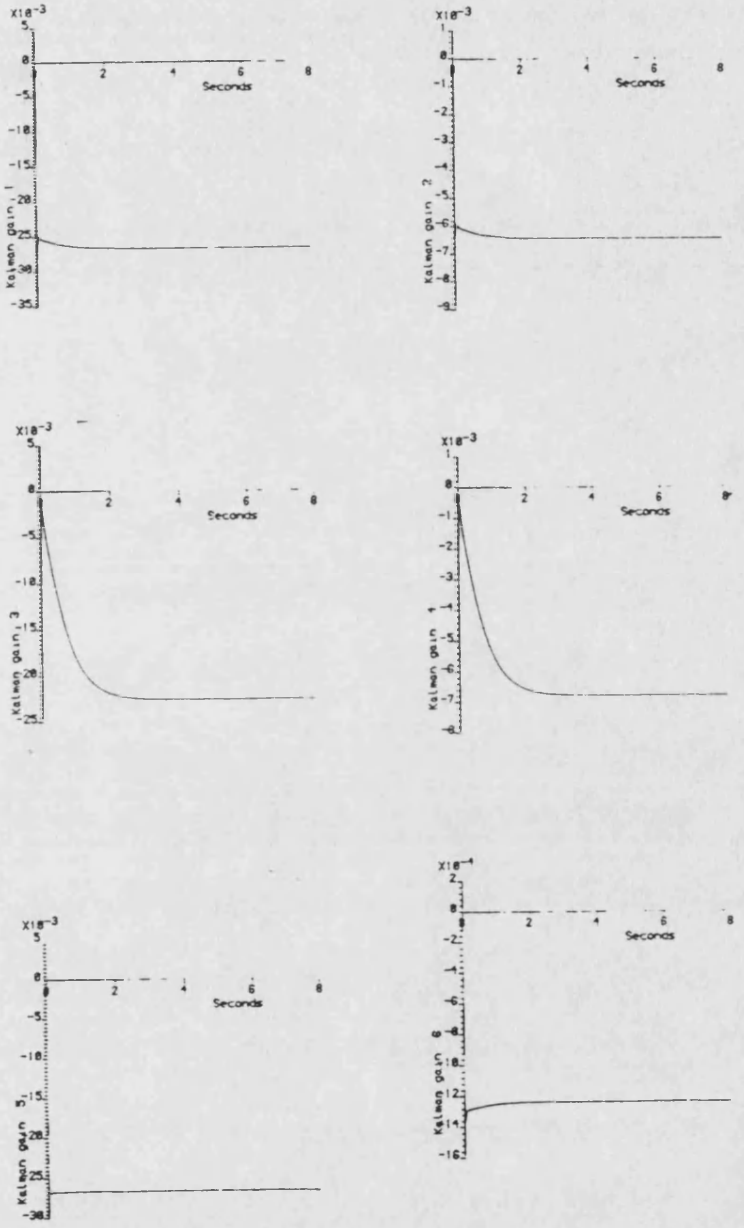


Figure 8.17: Time histories of first six Kalman filter gains for nominal model

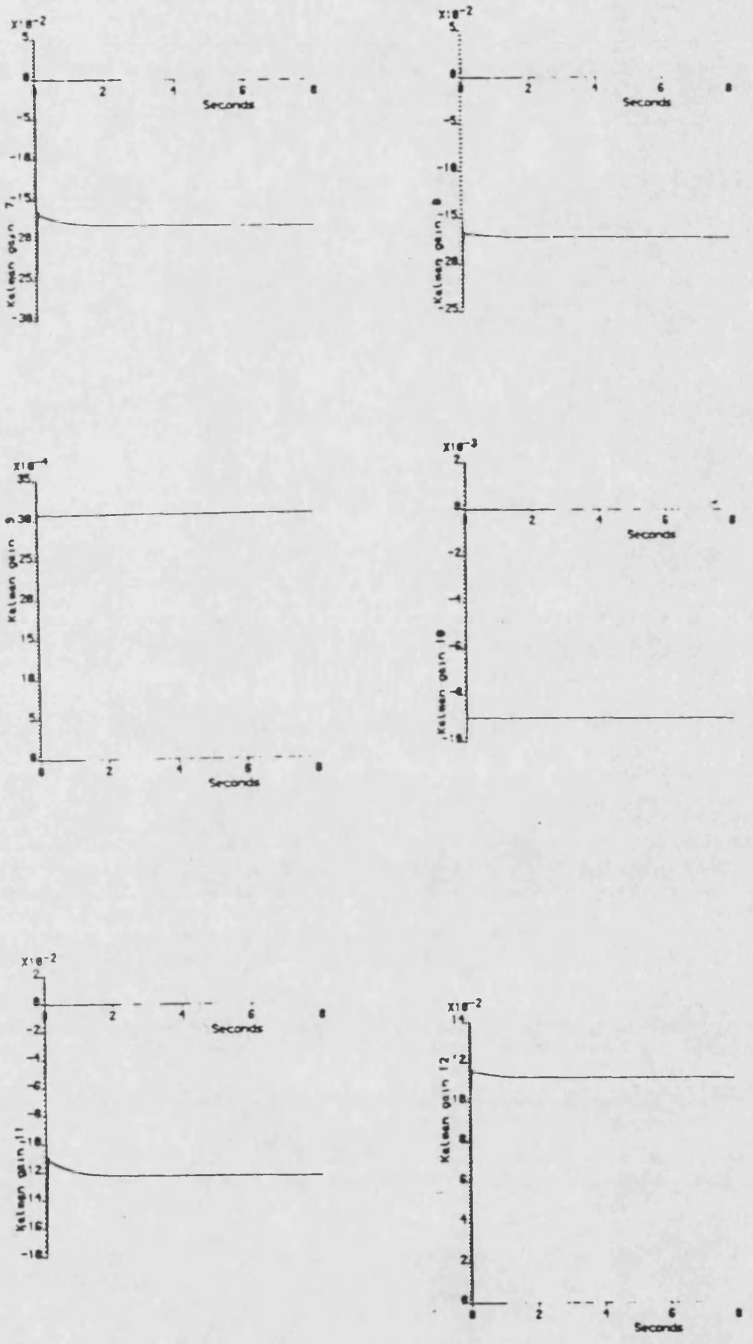


Figure 8.18: Time histories of second six Kalman filter gains for nominal model

step responses are slower and more highly damped than the case with the full order state estimator. Although, by redesigning the full order state estimator for a suitable set of pole locations, no doubt this type of response could be obtained. The filter gains time histories for the perturbed model no. 1 and the perturbed model no. 2 are shown in figures 8.20 and 8.21, and figures 8.23 and 8.24, respectively. It is interesting to note that for all these cases the dynamics of the filter gains are faster than the system responses, and also that the gains themselves tend towards steady-state values which differ between the cases. This demonstrates how the Kalman filter tends to adjust its characteristics to suit the conditions.

To examine the effects of the assumed values for the matrices describing the noise terms, a series of three simulation runs using the 6th order Kalman filter with the 10th order nominal model utilising different noise covariance matrices was carried out. The step responses and filter gains shown in figures 8.25, 8.26, and 8.27, correspond to the case when the matrix Q is a unit matrix, and the matrix R is a unit matrix scaled by 10. The step responses and filter gains shown in figure 8.28, 8.29, and 8.30, correspond to the case when both matrix Q and matrix R are unity. The step responses and filter gains shown in figures 8.31, 8.32, and 8.33, correspond to the case when the matrix Q is a unit matrix scaled by 10 and matrix R is a unit matrix.

It can be seen that there is no noticeable difference between the step responses, nor in the dynamic responses of the filter gains, but as would be expected, the steady-state filter gains are different between the three cases.

As a consequence of these results, a steady-state Kalman filter was investigated to ascertain what level of performance degradation would occur if the gains were fixed at their steady-state values. Of course, one way of obtaining the steady-state filter gains is via a simulation of the Kalman filter. However, this is computational expensive, particularly for high order systems, compared with other methods (see Chapter 5). Thus the program WIENER was used to compute the steady-state Kalman filter gains. It should be noted that the name of this program may be misleading, as although the steady-state Kalman filter is essentially very similar to a multivariable Wiener filter, the algorithm used to compute the gains has no connection with the algorithms originally proposed by Wiener [85].

Again, the effects of varying the weighting matrices Q and R was examined, this time over a range of 1:1 to 1:100, and these results are shown in table 8.9. The case of unit Q and R matrices was chosen for further investigation.

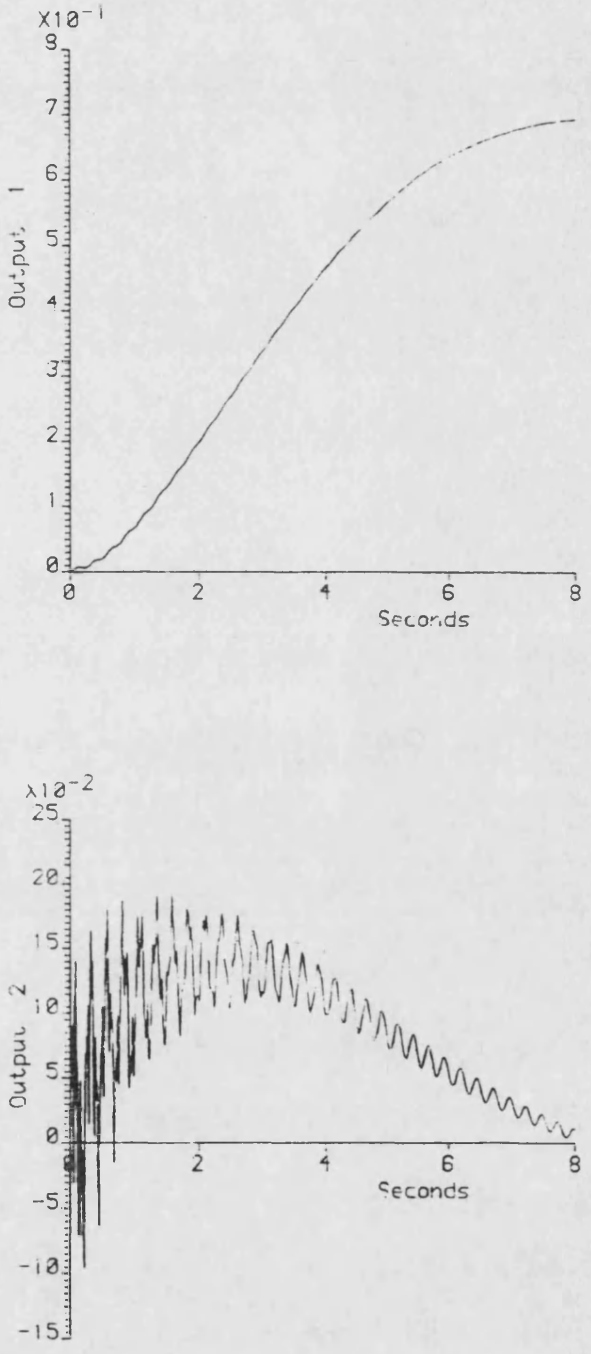


Figure 8.19: Step responses of perturbed model no. 1 with Kalman filter

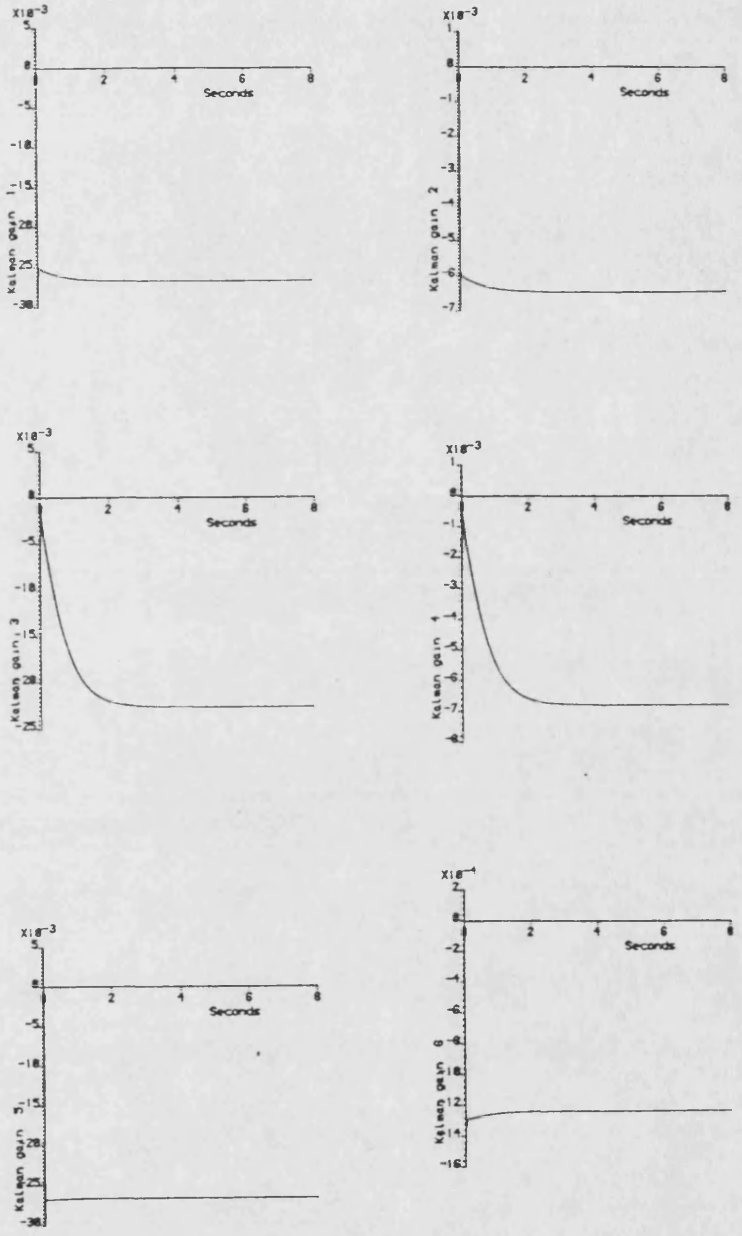


Figure 8.20: Time histories of first six Kalman filter gains for perturbed model no. 1

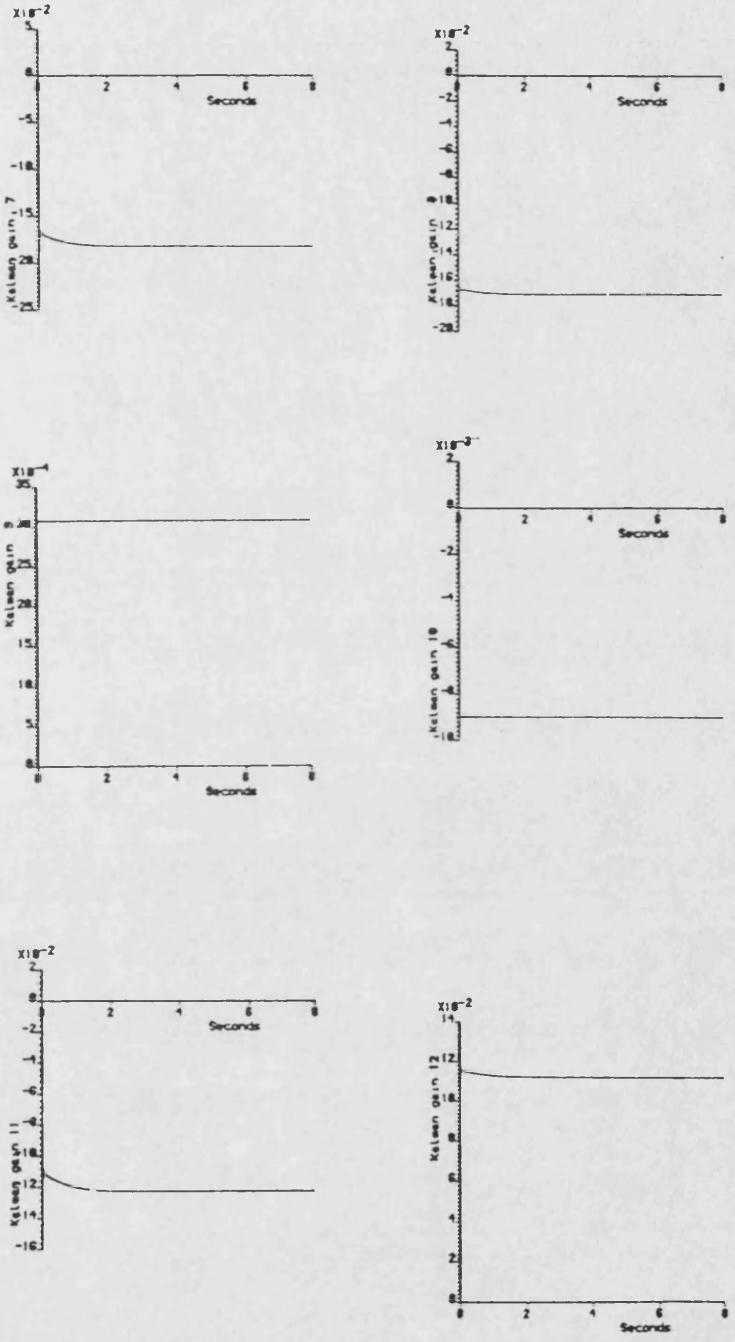


Figure 8.21: Time histories of second six Kalman filter gains for perturbed model no. 1

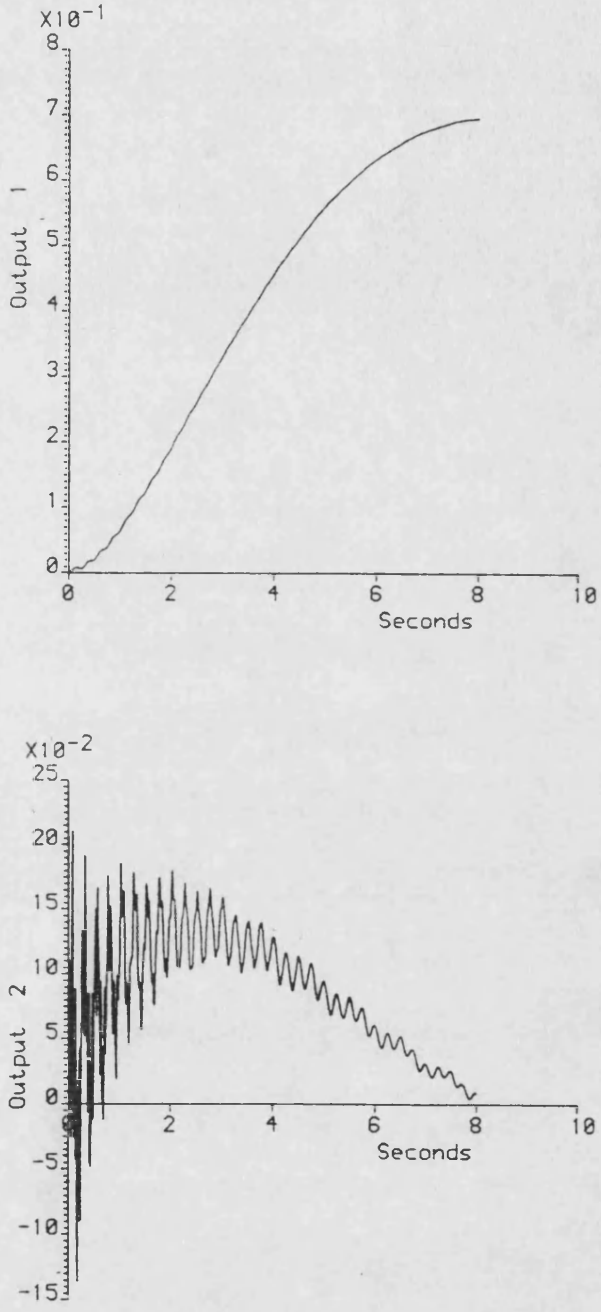


Figure 8.22: Step responses of perturbed model no. 2 with Kalman filter

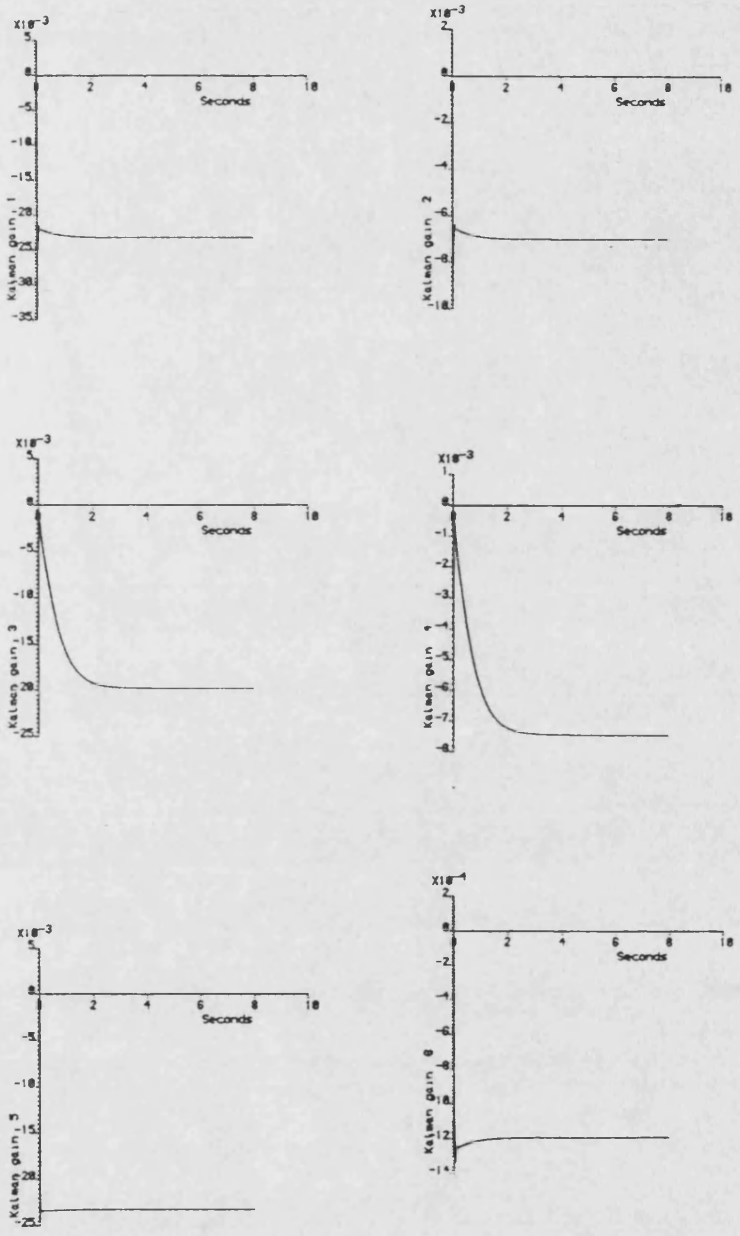


Figure 8.23: Time histories of first six Kalman filter gains for perturbed model no. 2

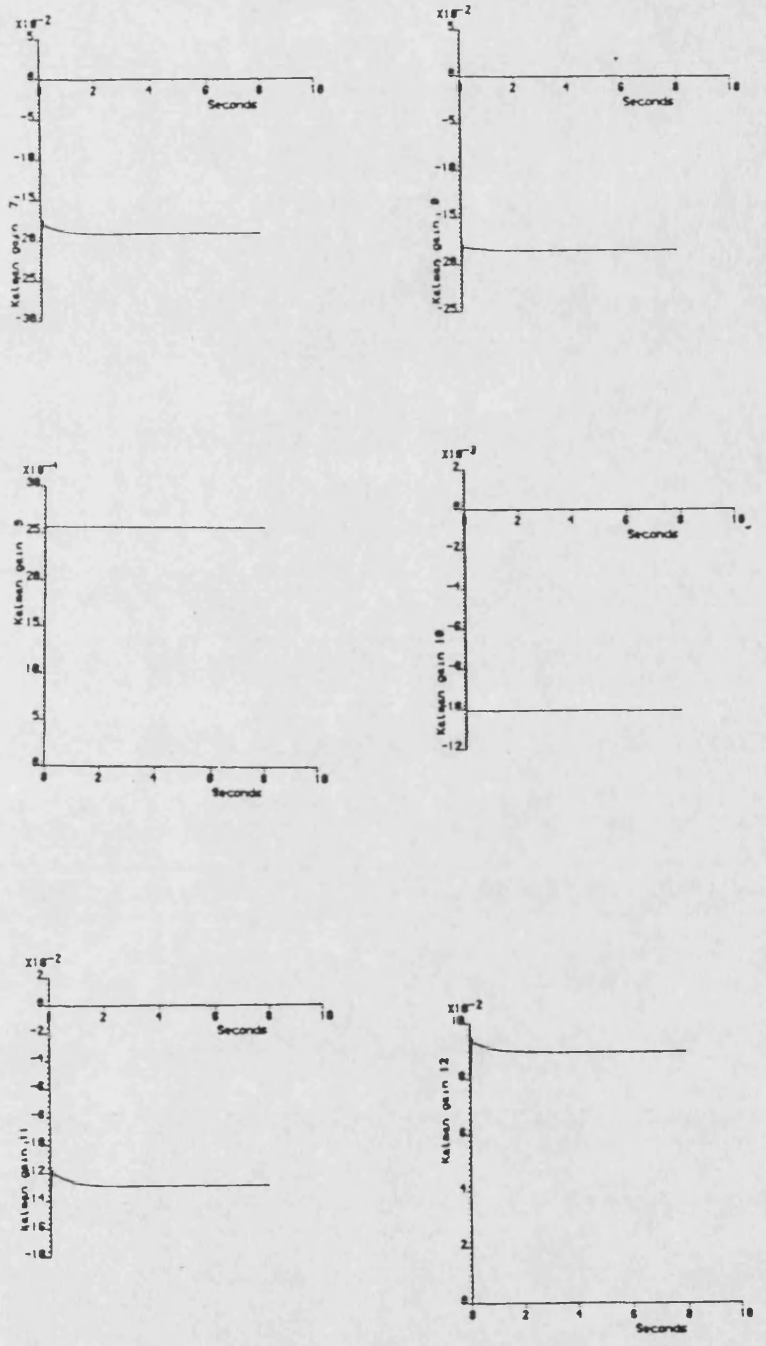


Figure 8.24: Time histories of second six Kalman filter gains for perturbed model no. 2

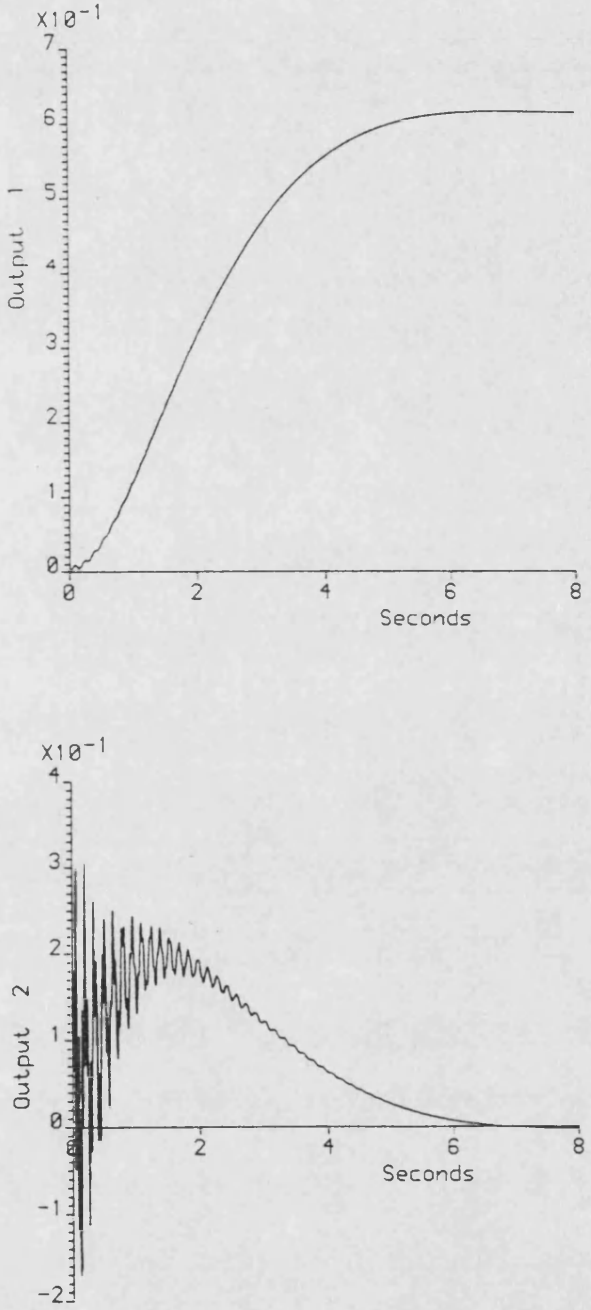


Figure 8.25: Step responses of nominal model with Kalman filter (q/r=0.1)

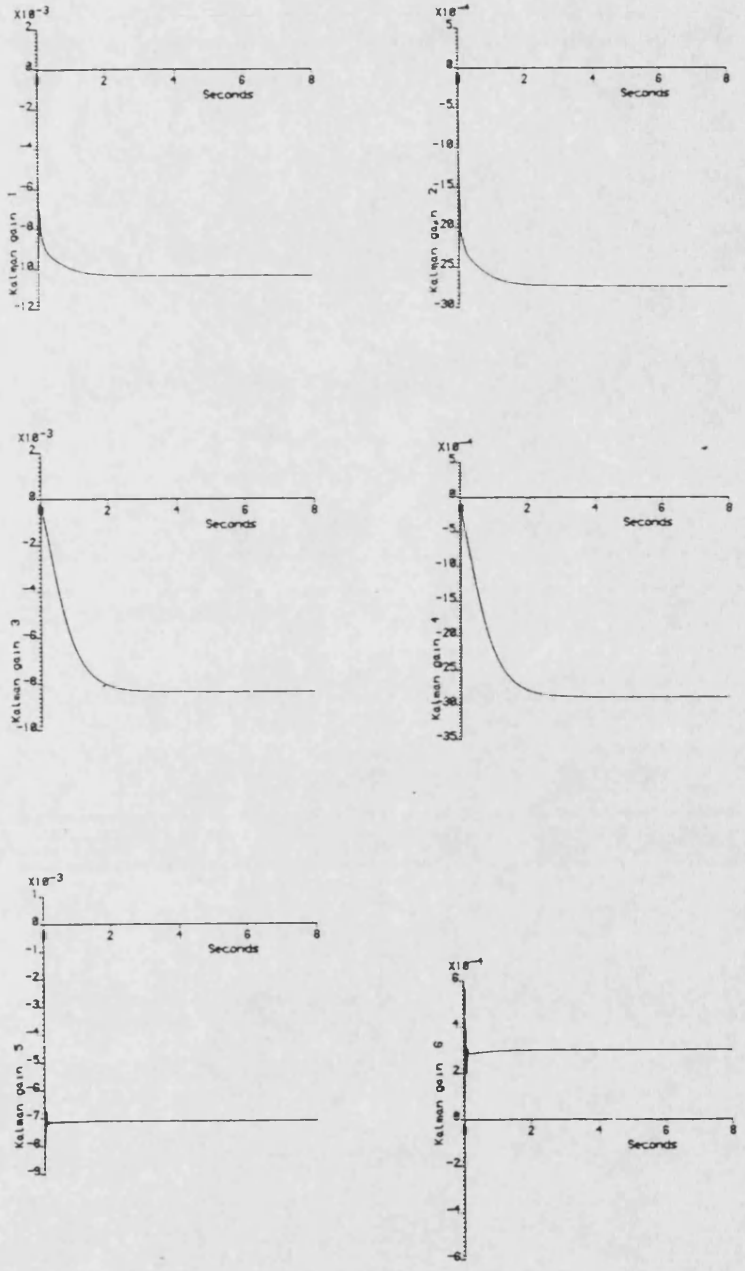


Figure 8.26: Time histories of first six Kalman filter gains for nominal model (q/r=0.1)

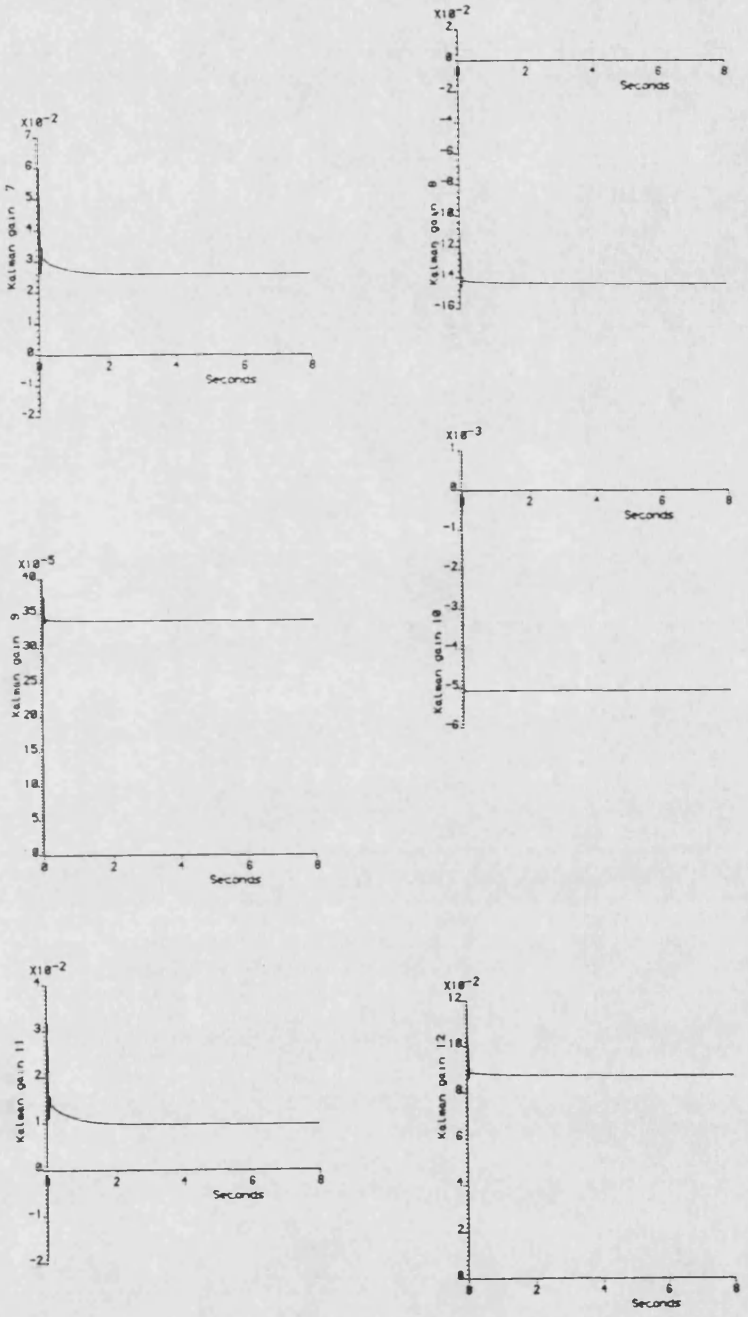


Figure 8.27: Time histories of second six Kalman filter gains for nominal model (q/r=0.1)

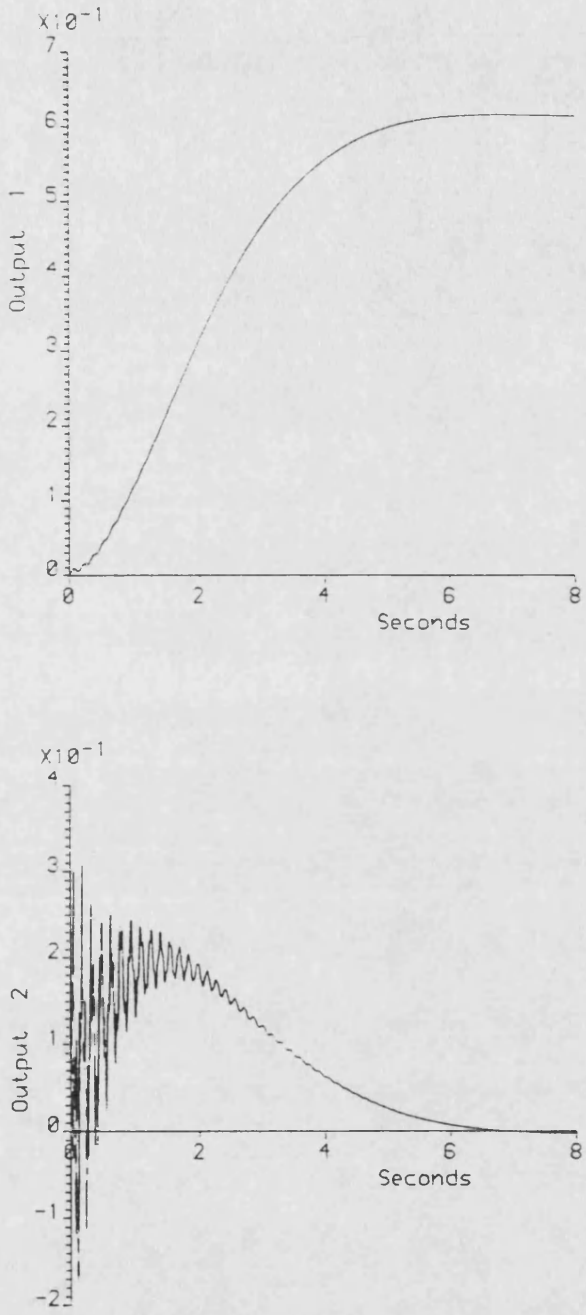


Figure 8.28: Step responses of nominal model with Kalman filter (q/r=1.0)

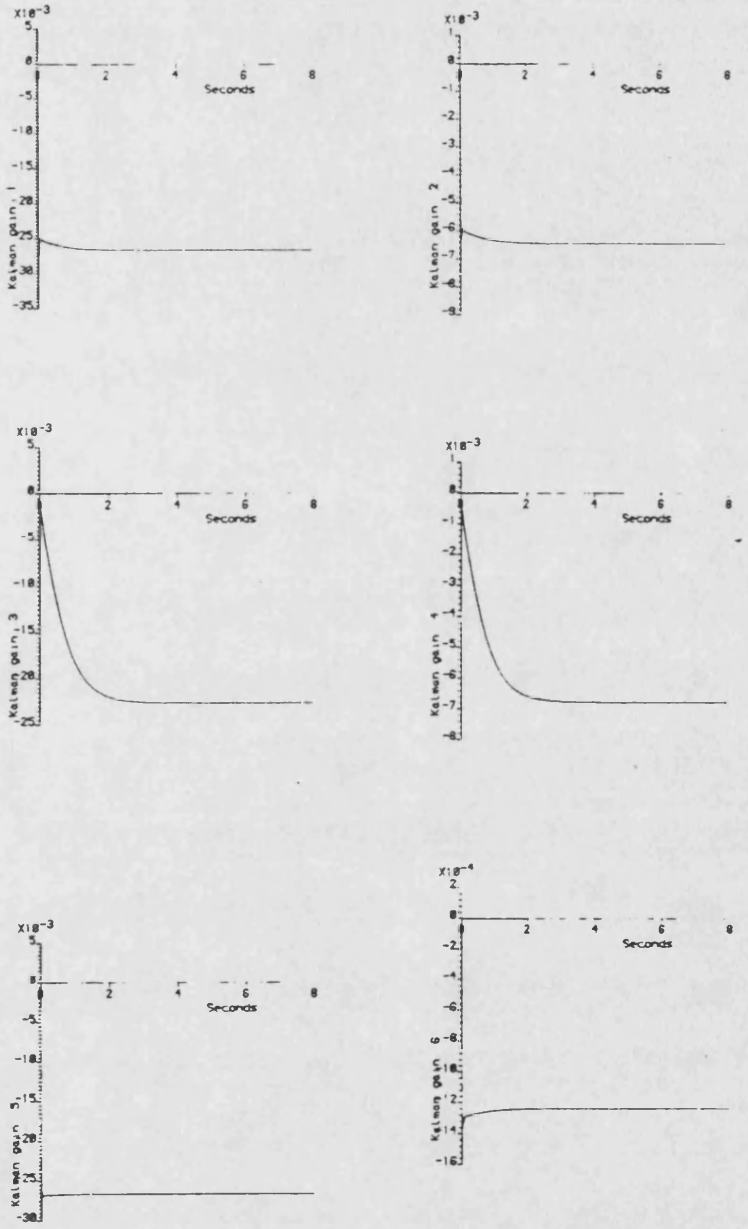


Figure 8.29: Time histories of first six Kalman filter gains for nominal model (q/r=1.0)

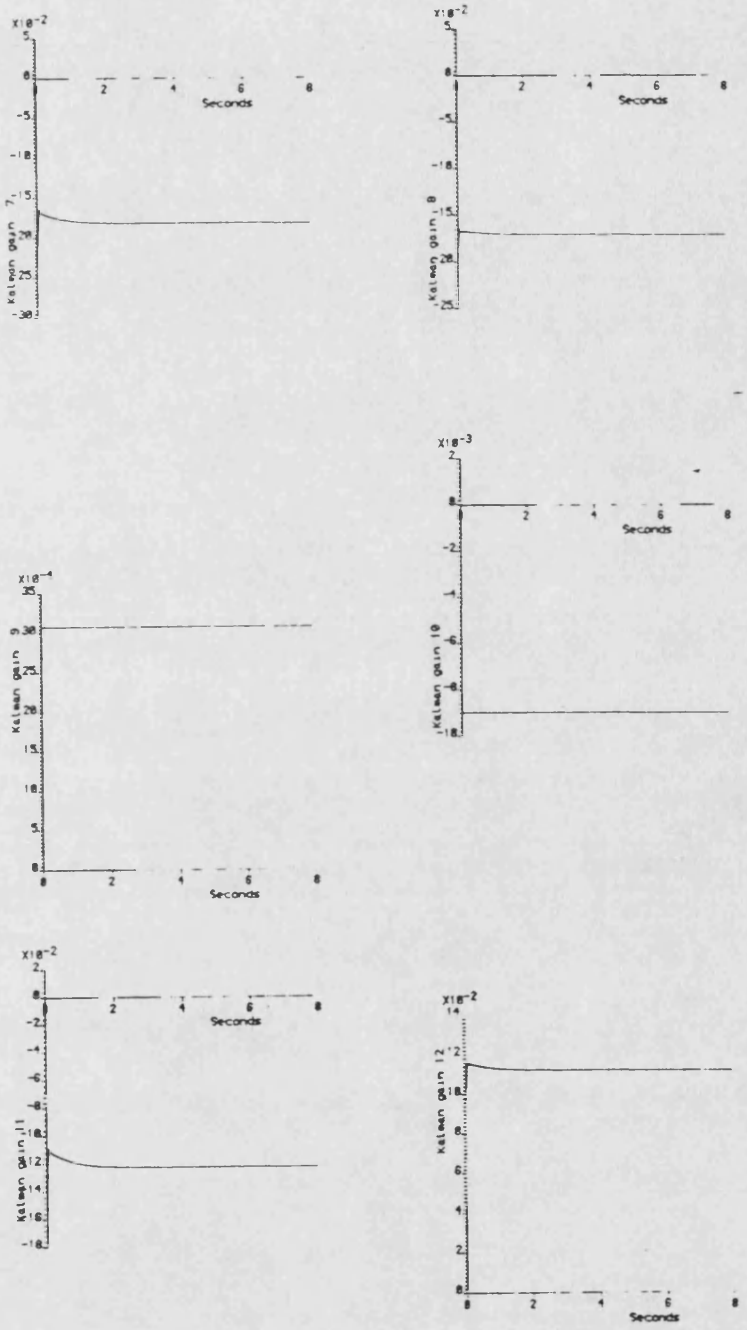


Figure 8.30: Time histories of second six Kalman filter gains for nominal model (q/r=1.0)

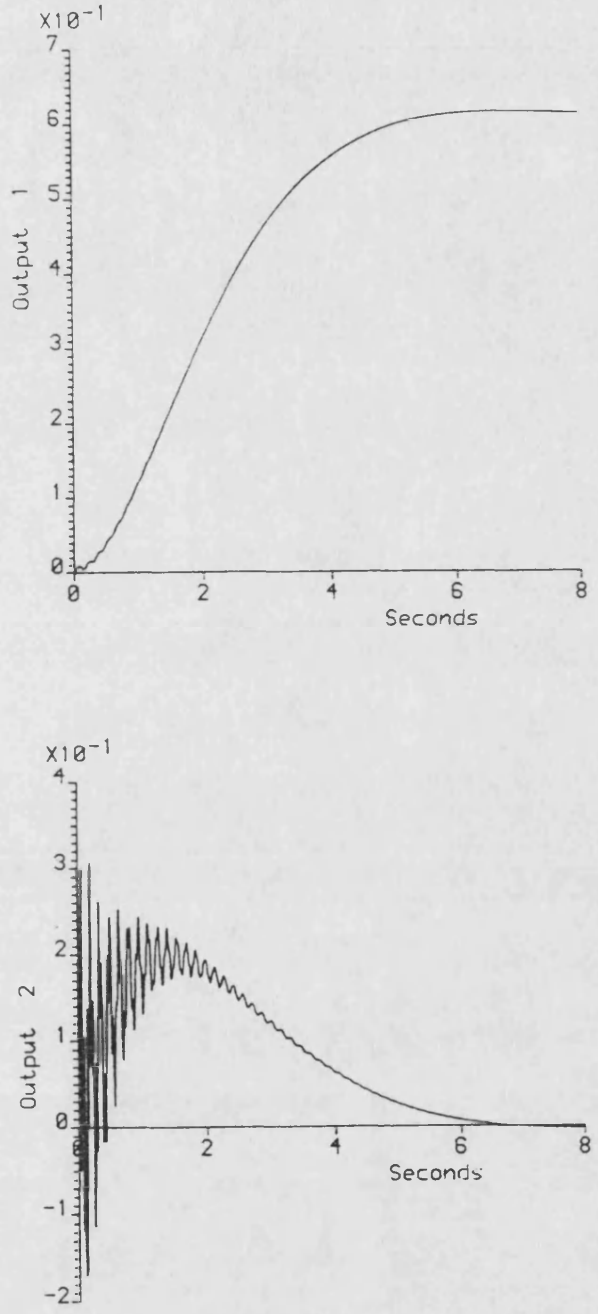


Figure 8.31: Step responses of nominal model with Kalman filter (q/r=10)

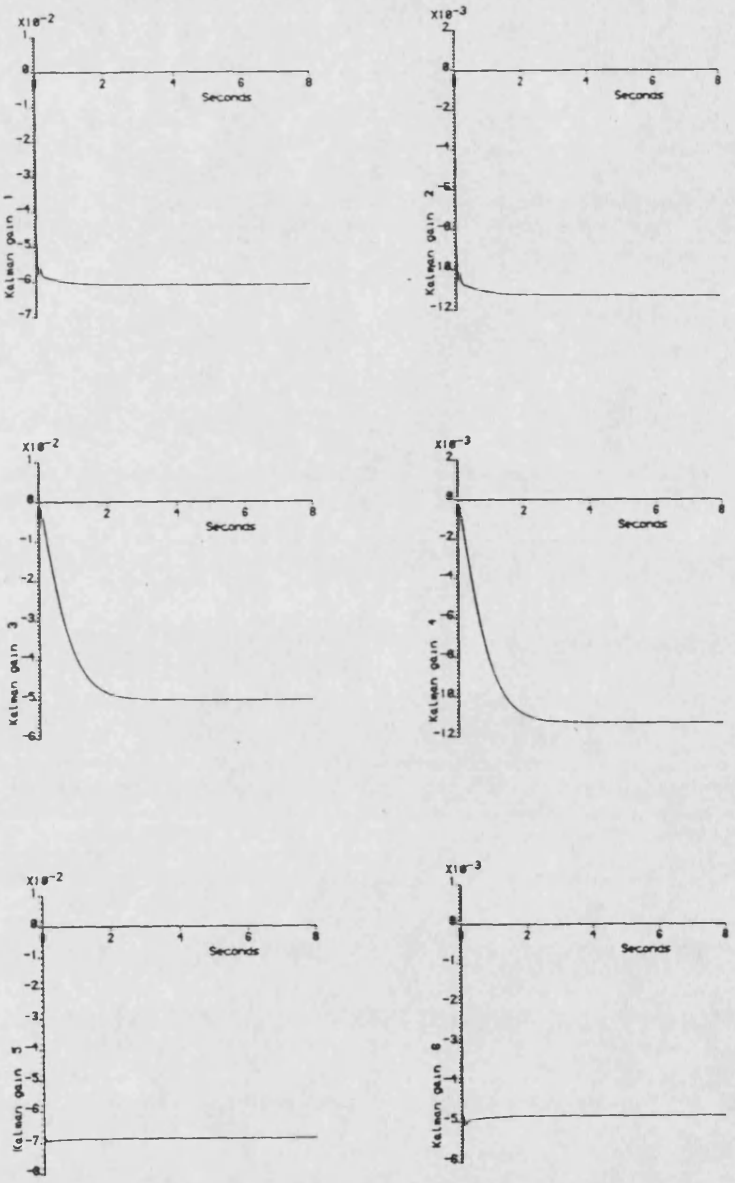


Figure 8.32: Time histories of first six Kalman filter gains for nominal model (q/r=10)

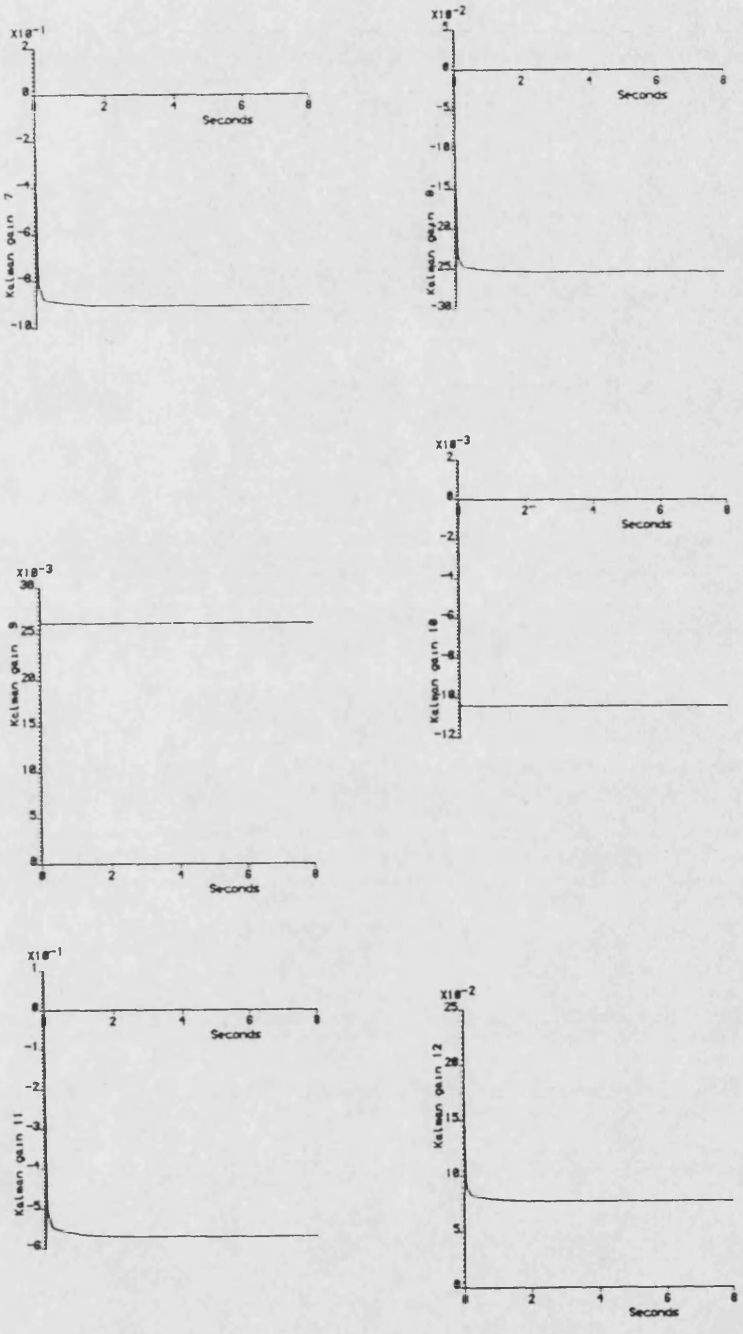


Figure 8.33: Time histories of second six Kalman filter gains for nominal model (q/r=10)

q/r = 0.01	Gain Matrix	$\begin{bmatrix} -0.56129 & -0.09029 \\ -0.09494 & -0.03140 \\ -0.00242 & 0.00068 \\ 0.02421 & -4.03170 \\ 0.00070 & -0.02692 \\ -0.01311 & 13.3710 \end{bmatrix}$
	Eigenvalues	$-29.214 + j\ 145.4$ $-29.214 - j\ 145.4$ $-4.9059 + j\ 42.41$ $-4.9059 - j\ 42.41$ $-0.1792 + j\ 0.170$ $-0.1792 - j\ 0.170$
q/r = 0.1	Gain Matrix	$\begin{bmatrix} -1.02220 & -0.18799 \\ -0.29515 & -0.11352 \\ -0.01285 & 0.01405 \\ 0.22484 & -12.9910 \\ 0.00301 & -0.19414 \\ -0.07993 & 35.1390 \end{bmatrix}$
	Eigenvalues	$-81.826 + j\ 164.2$ $-81.826 - j\ 162.2$ $-9.4519 + j\ 43.44$ $-9.4519 - j\ 43.44$ $-0.3295 + j\ 0.291$ $-0.3295 - j\ 0.291$
q/r = 1.0	Gain Matrix	$\begin{bmatrix} -2.01860 & -0.34112 \\ -0.94970 & -0.31316 \\ -0.11211 & 0.12812 \\ 1.78530 & -40.6700 \\ 0.01591 & -0.87823 \\ -0.05354 & 67.2350 \end{bmatrix}$
	Eigenvalues	$-183.29 + j\ 231.9$ $-183.29 - j\ 231.9$ $-11.082 + j\ 44.07$ $-11.082 - j\ 44.07$ $-0.6343 + j\ 0.458$ $-0.6343 - j\ 0.458$

Table 8.9: Effects of varying weighting matrices Q and R on steady-state Kalman filter gains

The application of the controller formed by the state feedback matrix and the 6th order steady-state Kalman filter to the various models was simulated. The step responses of the 6th order design model with this controller are shown in figure 8.34, where it can be seen that the responses are essentially the same as those of the time-varying Kalman filter, as might be expected.

The responses of the 10th order nominal model with this controller are shown in figure 8.35, and again, as would be expected from earlier results, there is no noticeable deterioration in performance from the time-varying case.

The cases of the perturbed model no. 1 and perturbed model no. 2 are shown in figures 8.36 and 8.37, respectively. Again, as might be expected bearing in mind earlier results, the responses have slowed, but are very well damped. This suggests that a steady-state Kalman filter is likely to give almost as good a performance as a time-varying Kalman filter for this particular problem, with considerable savings in computational requirements (see Chapter 6).

8.3.2 Controller Design—Controller Order Reduction

The second approach follows the high order controller design—controller reduction philosophy. Initially, a controller was designed for the 10th order nominal model, which obviously is also 10th order, and then the program QCOVER was used to attempt to derive a suitable reduced-order controller with similar dynamic properties.

The 10th order controller was designed using the programs LQREG and WIENER, using unity weighting matrices in both cases. The responses of the nominal model to this controller are shown in figure 8.38.

The program QCOVER was then used to attempt to produce controllers of lower order. As has already been mentioned (see Chapter 7), the program QCOVER starts by obtaining an "equivalent" first order controller, then repeats the procedure incrementing the order of the resulting controller each time until the dimension of the full order controller is reached. The program was run several times using an output covariance matrix which was a scaled unit matrix, with the scaling adjusted through the range 0.0001 to 10. However, in all cases the program terminated prematurely due to numerical overflow after deriving the 6th order controller. Also, the variation between the reduced order controllers caused by the variation of the output covariance matrix was minimal. Consequently, the results of all these runs are not shown, only the results for the case of a unit output covariance matrix are

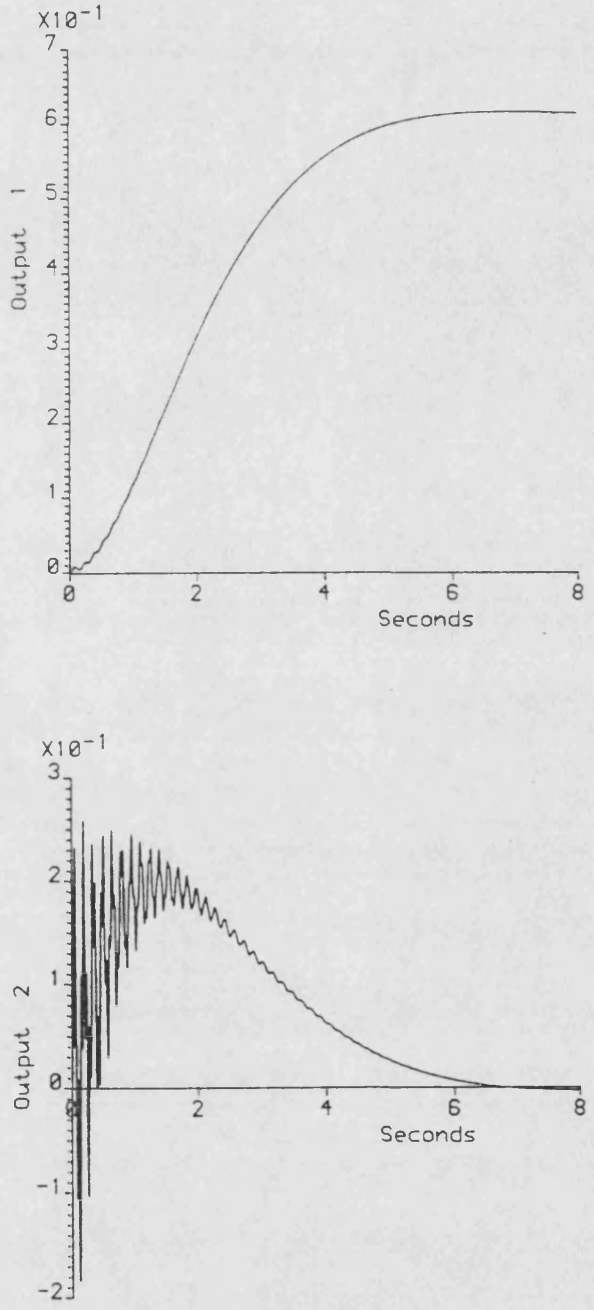


Figure 8.34: Step responses of design model with steady-state Kalman filter

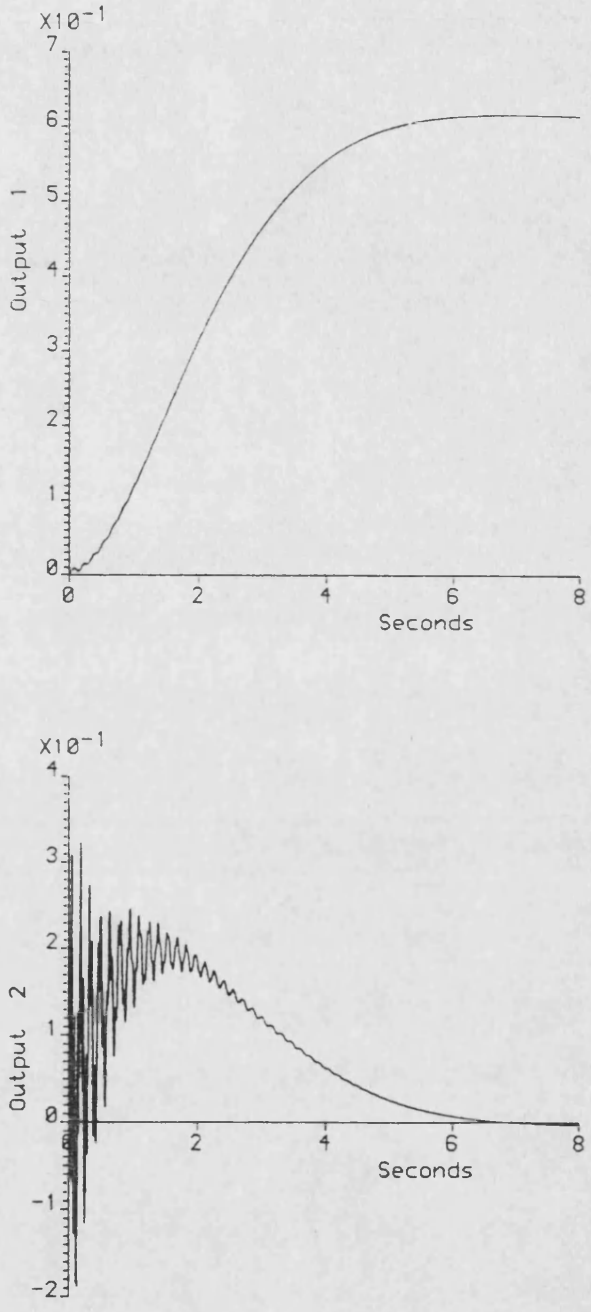
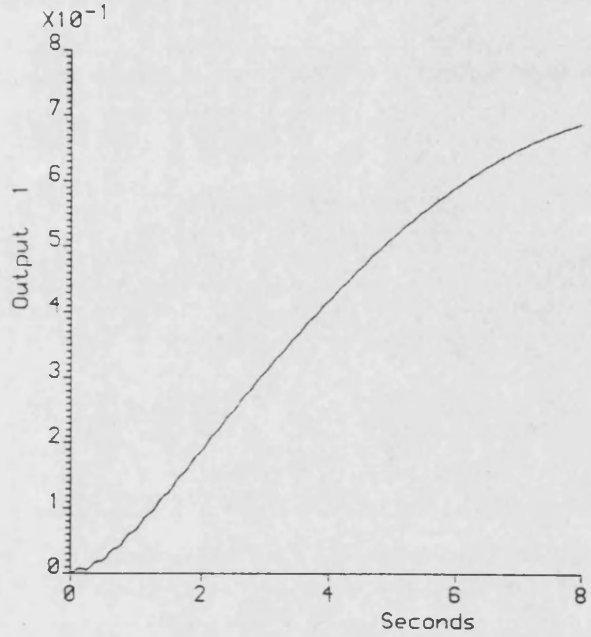


Figure 8.35: Step responses of nominal model with steady-state Kalman filter



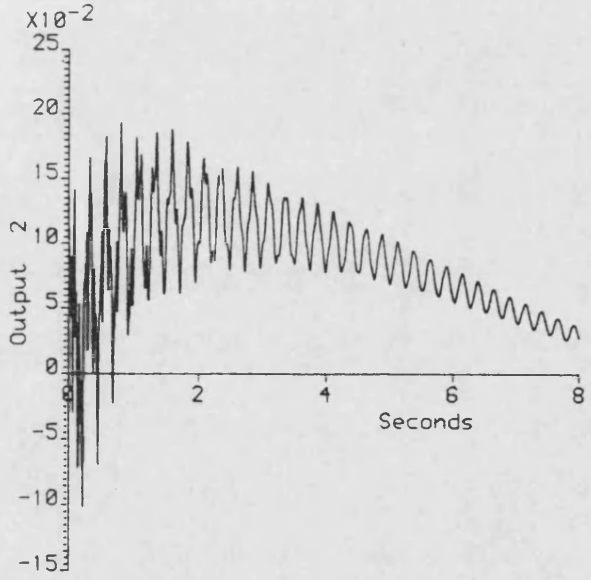


Figure 8.36: Step responses of perturbed model no. 1 with steady-state Kalman filter

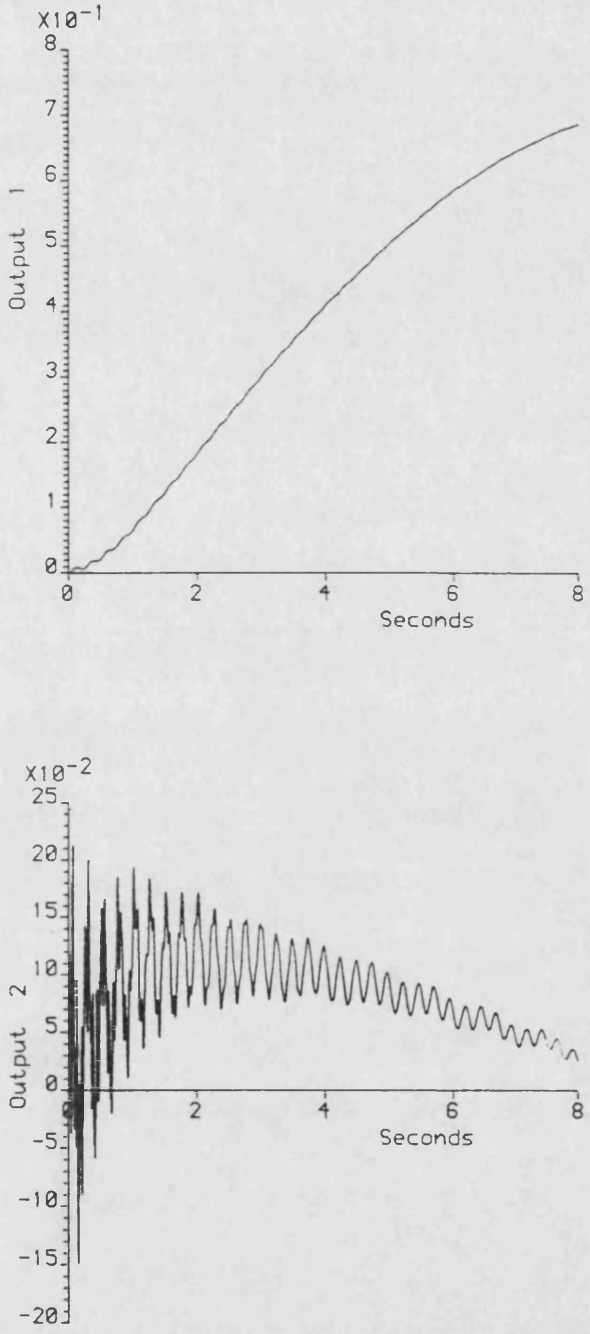


Figure 8.37: Step responses of perturbed model no. 2 with steady-state Kalman filter

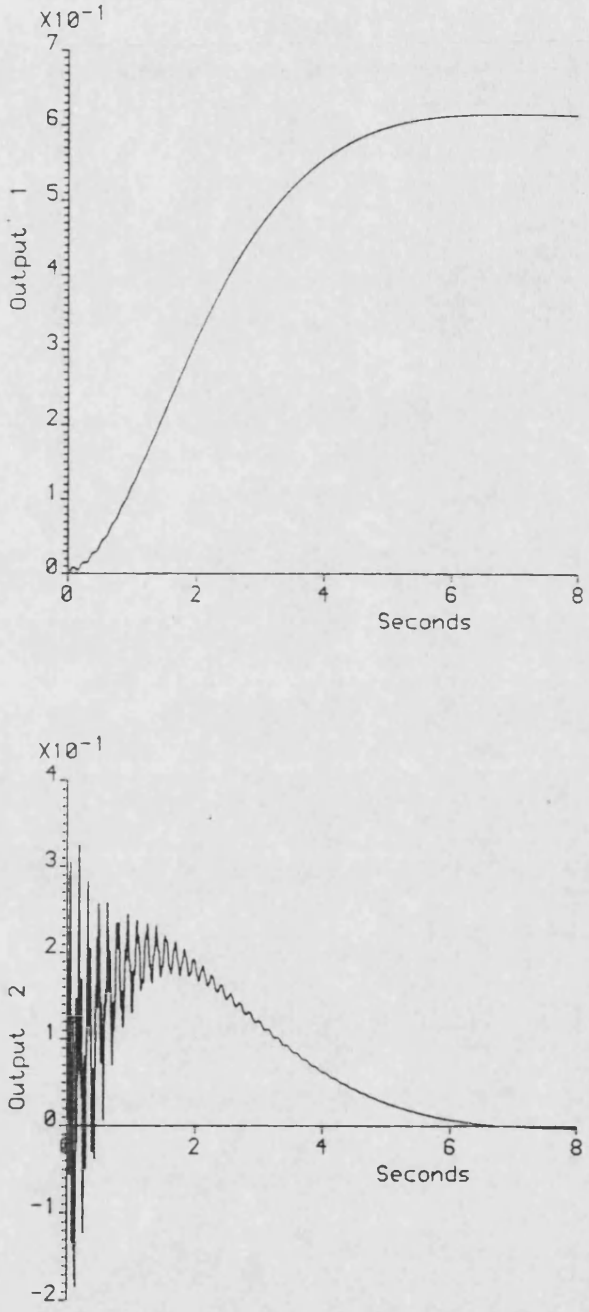


Figure 8.38: Step responses of nominal model with full order controller

shown in table 8.10.

These results, unfortunately, are not particularly useful, because the reduced order controllers either result in unstable closed loop systems, or if stable, have very slow dominant poles. The most promising result is the 5th order controller, and thus the behaviour of the 10th order nominal model subject to this controller was simulated. The step responses shown in figure 8.39 show that the performance is totally unacceptable, being much too slow and poorly damped.

8.4 Output Feedback Based Controllers

The controllers considered in the following sections generate feedback signals from the output measurements directly through fixed gains, thus there are no dynamic models of the system within the controller structure. Two particular techniques, which have already been described in detail in Chapter 5, are used to attempt to design such a controller.

8.4.1 Input–Output Decoupling (IMSC)

The program IMSC was used to compute an output feedback matrix by attempting to decouple the input and output matrices (the Independent Modal Space Control technique, see Chapter 5). The desired pole locations were chosen to be the same as those produced by the program LQREG (see Section 8.3.1), to enable comparisons to be drawn. The results of the program IMSC using the 10th order nominal model are shown in table 8.11, where it can be seen that the eigenvalues of the resulting closed loop system are not even close to the desired locations. However, as the feedback terms are positive (thus the feedback matrices are positive-definite), from the theory detailed in Chapter 4 and Appendix A.2, the overall system will be stable. The simulated step responses are shown in figure 8.40, and indeed confirm that the rigid mode response is much too slow and poorly damped, hence this controller is of little value.

8.4.2 Rigid–Mode Design

This controller design method is based on the stability properties of positive-definite feedback matrices as discussed in Chapter 4 and Appendix A.2. The program RMDES was used to place the poles of the rigid mode in the required locations (again, the locations defined by the program LQREG

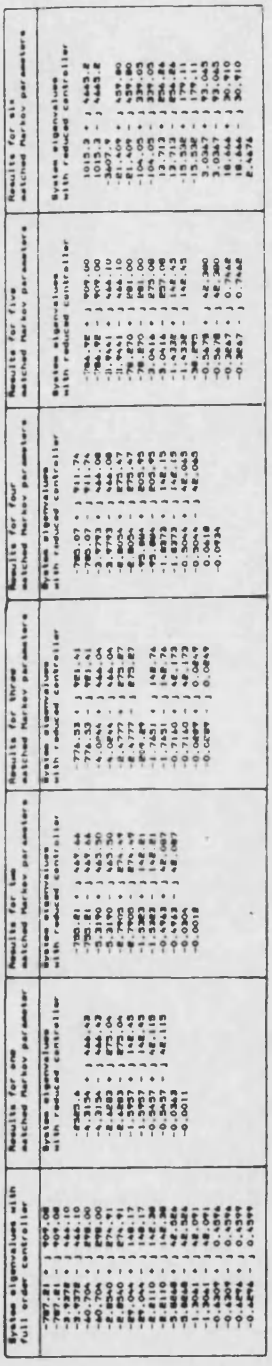
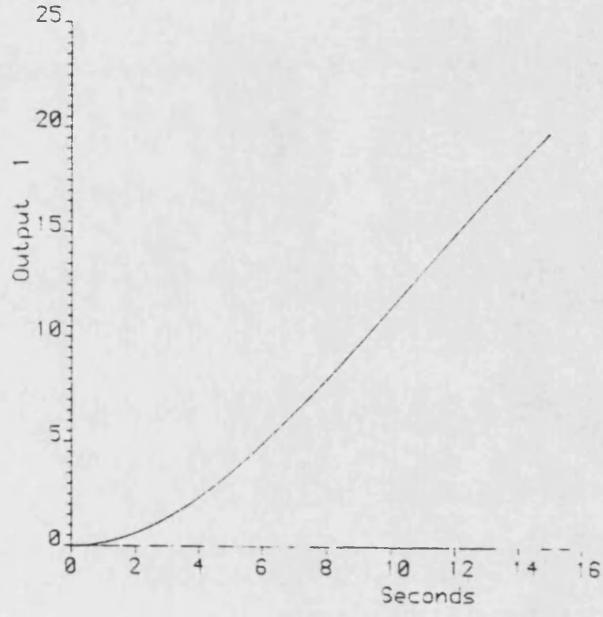


Table 8.10: Numerical results of program QCOVER for 10th order beam controller



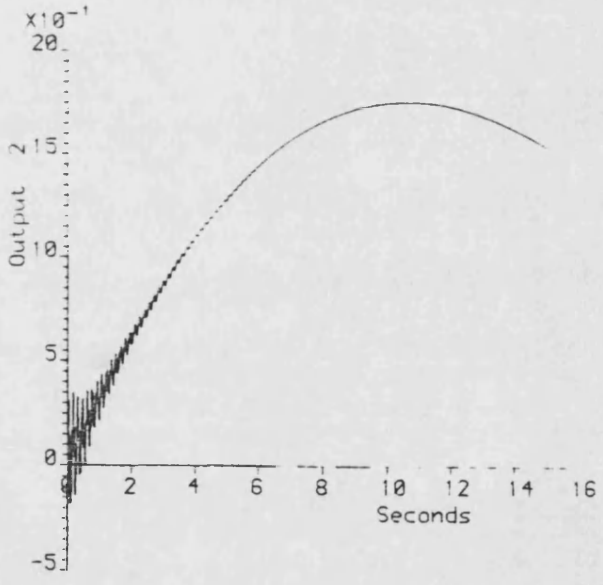


Figure 8.39: Step responses of nominal model with 5th order QCOVER controller

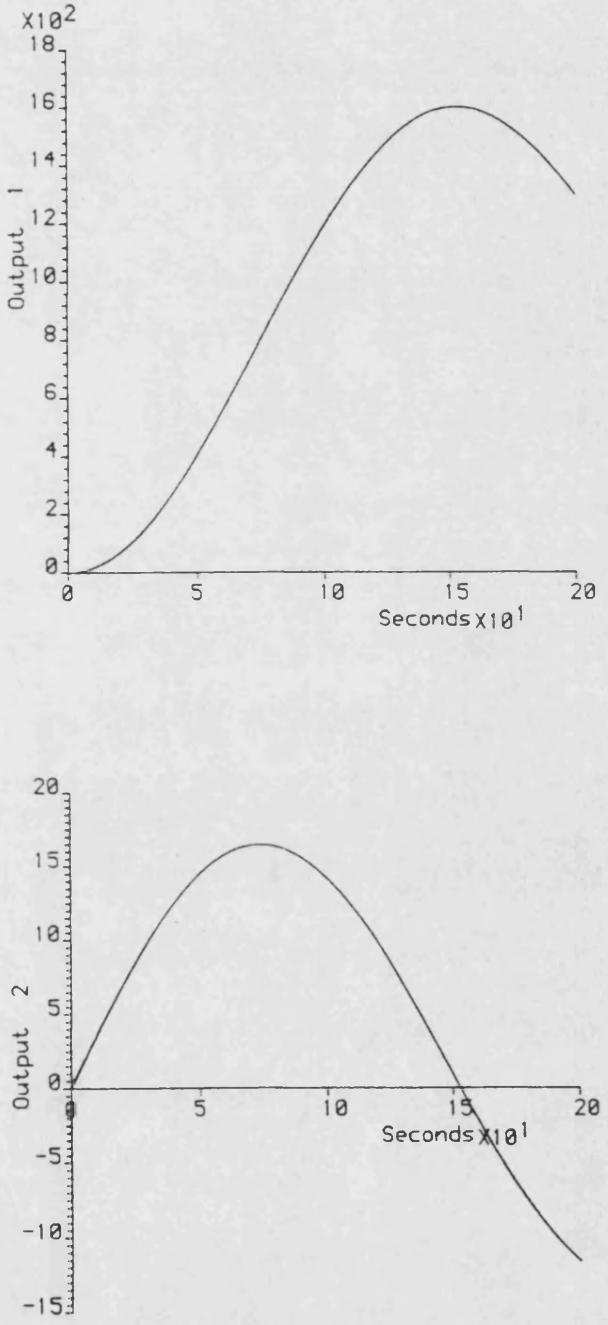


Figure 8.40: Step responses of nominal model with IMSC controller

Gain Matrix	[0.001146 0.006432]
Eigenvalues	-0.76839 + j 142.40 -0.76839 - j 142.40 -0.23137 + j 42.110 -0.23137 - j 42.110 -0.00119 + j 0.02055 -0.00119 - j 0.02055

Table 8.11: Results of IMSC design

were used, see Section 8.3.1), and by constraining the feedback matrices to be positive-definite (which in this case can be done by ensuring that the two terms are both positive), a stable system can be guaranteed. The results of the program are shown in table 8.12, where it can be seen that the design was successful. This is confirmed by the simulated step responses of the 10th order nominal model with this controller which are shown in figure 8.41. The step responses of the perturbed model no. 1 and the perturbed model no. 2 with this controller are shown in figure 8.42 and 8.43, respectively, where it can be seen that only slight deterioration of the performance is noticeable.

8.5 Comparative Remarks

The technique of using model reduction and designing a controller based on the reduced order model appears to work well. The technique of modal cost analysis (see Chapter 3) as a method of model reduction is convenient to use, and good control can be exercised by the analyst in producing reduced order models. It also provides a means of judging the validitity of the reduced order models derived via the model error expression.

Treating the design problem as a linear quadratic regulator problem (see Chapter 5) is a convenient means of computing a state feedback matrix. Various performance parametrics can be adjusted by a suitable choice of weighting matrices. The actual choice of weighting terms is difficult, if not impossible, to define *a priori*. Thus an iterative approach generally has to be adopted, where the nominal weighting terms are chosen initially in a

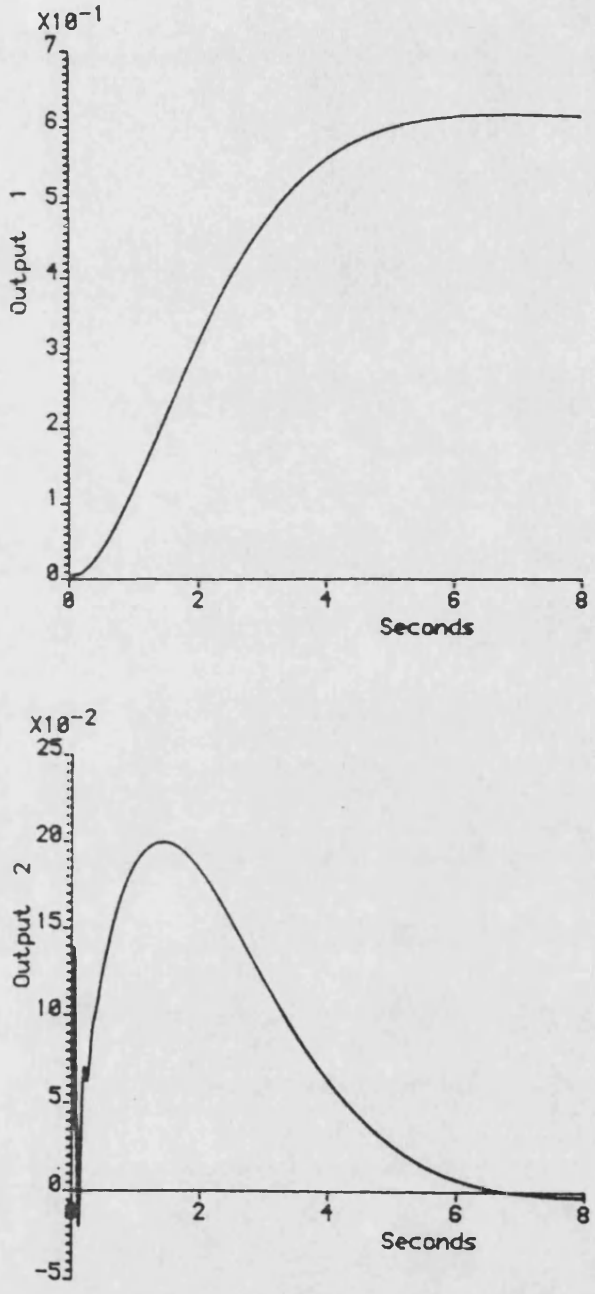


Figure 8.41: Step responses of nominal model with rigid mode controller

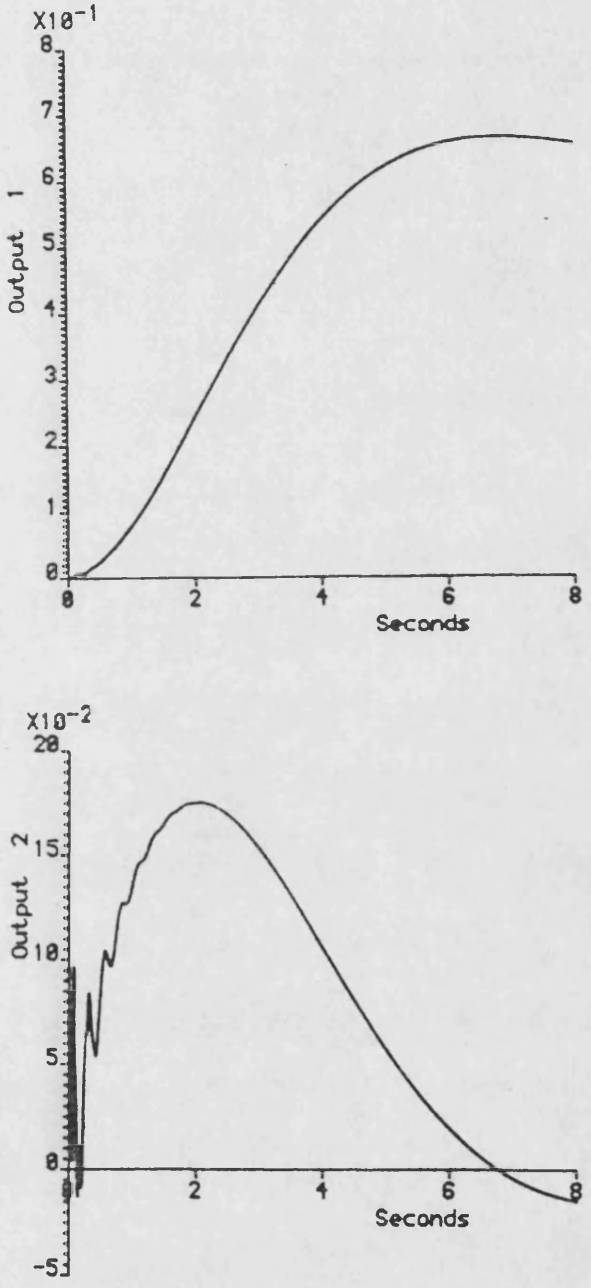


Figure 8.42: Step responses of perturbed model no. 1 with rigid mode controller

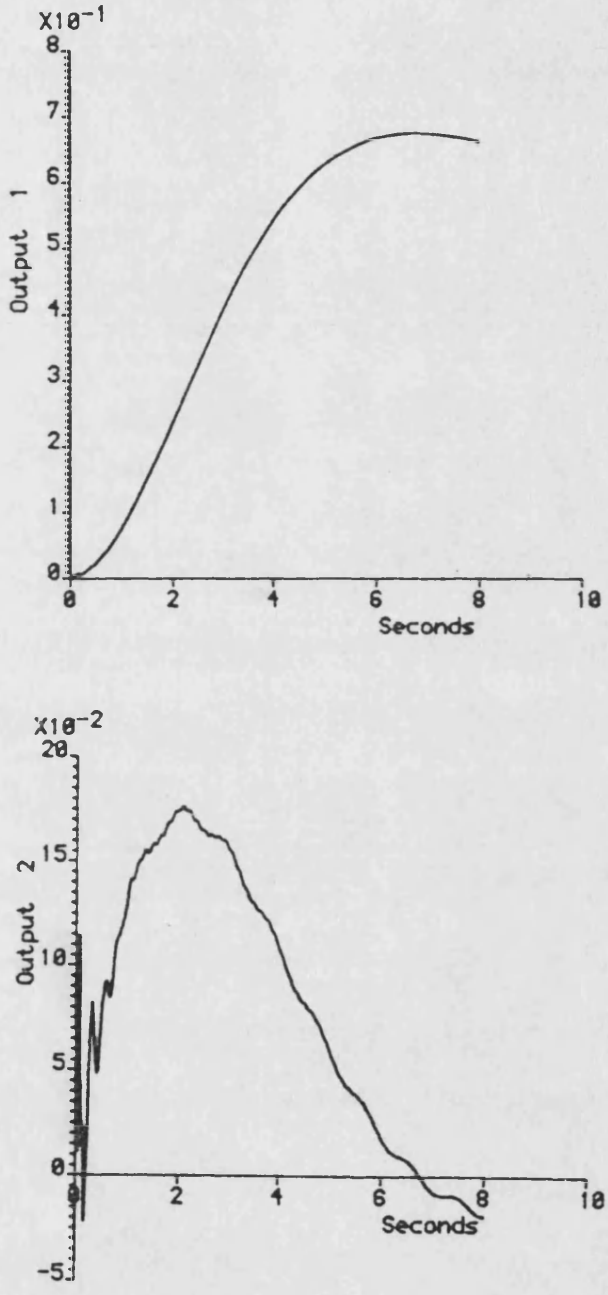


Figure 8.43: Step responses of perturbed model no. 2 with rigid mode controller

Gain Matrix	[1.643 3.403]
Eigenvalues	-0.62897 + j 0.46015 -0.62897 - j 0.46015

Table 8.12: Results of rigid mode design

rather arbitrary manner (say as unit matrices), and the design is carried out. The characteristics of the resulting closed loop system can be adjusted by judicious alteration of the weighting terms, based on the guidelines described in Chapter 5, and repeating the design process. Obviously, this procedure is repeated until a satisfactory design has been obtained.

The problem of designing a state estimator to provide estimates of the states required for feedback then remains. Essentially three types of dynamic state estimator have been considered, a full order state estimator, a minimal order state estimator, and a Kalman filter.

The design of a full order state estimator was carried out using a pole placement algorithm via the observable canonical form. It should be noted, however, that whilst this algorithm functioned perfectly well in this case, it is prone to numerical problems with higher order systems (see Chapter 5), and thus its use for more complex cases is likely to result in failure. The resulting design in this case worked well, even in the presence of considerable model perturbation.

The design of a minimal order state estimator was also carried out using a pole placement algorithm, so the comments above concerning the application to high order problems also apply to this case. The resulting design in this case performed well when the model mismatch was small, but the perturbed models resulted in unstable closed loop systems, suggesting that the minimal order state estimator is more sensitive to model mismatch than the full order state estimator. These comments agree with the general understanding about the design of robust (that is insensitive to parameter variation) multivariable systems (see for example [113]), which suggests that the greater the level of redundancy in the controller, then in general the greater the degree of robustness of the system.

A Kalman filter largely designs itself, as it updates the gain matrix

according to covariance computations. Only limited influence on the steady-state gains is available via the noise term matrices, which have to be defined before the filter is utilised.

The closed loop system utilising a Kalman filter for state estimation performed well, even in the presence of model mismatch. The only apparent drawback to the use of a Kalman filter for state estimation appears to be the large real-time computational requirements compared to other forms of dynamic state estimator (see Chapter 6). As the essential structure of a Kalman filter is the same as that of the full order state estimator, this prompted the examination of the use of a steady-state Kalman filter, where the gain matrix is fixed at the steady-state values. Rather than use a simulation to obtain the steady-state filter gains, the gains were computed "off-line", which is generally much more efficient.

The steady-state Kalman filter performed almost as well as the time-varying Kalman filter, even in the presence of model mismatch. These results suggest that the computational advantage of the fixed gain filter can be obtained with only minor deterioration of the closed loop system performance from that of the time-varying filter. Some influence on the characteristics of the fixed gain filter is available by judicious choice of the noise matrix terms, in a similar manner to that described above for the linear quadratic regulator design method.

The technique of designing a controller for the high order system and reducing the order of the controller is currently more problematic. Firstly, the problem of designing a controller for the high order system is not completely straightforward. As has already been mentioned (see Chapter 5), almost all of the state estimator design methods examined suffer from numerical difficulties with high order systems, the only notable exception is the optimal filter method (the steady-state Kalman filter). Thus the choice of design method for the high order controller is severely restricted by the problem of numerical difficulties of the algorithms.

Secondly, having designed the high order controller, the problem of obtaining an "equivalent" controller which essentially maintains the closed loop characteristics produced by the high order controller then arises. The only algorithm that has been implemented to date for controller reduction was used, but the results are very disappointing. The algorithm got into numerical difficulties after producing about two-thirds of the range of expected controllers, and of those produced, several resulted in unstable closed loop systems, and only one exhibited any similarlity to the original controller, but its performance was sadly lacking. Thus it appears that the technique

of designing a high order controller and then reducing the order of that controller is currently impractical.

The design of controllers using output meaurements directly without attempting to reconstruct the states of the system is generally much simpler than the design of controllers which require the estimation of system states. The first technique examined, based on decoupling of the inputs and the outputs, is rather disappointing, the resulting system poles being quite a significant distance from the desired locations. This is reflected in poor performance of the closed loop system.

The second technique is based on the stability characteristics discussed in Chapter 4, and consequently, the elastic modes are completely ignored in the design process. However, the performance of the resulting controller is very good, increasing the level of damping of the elastic modes as well as giving the desired rigid mode response.

It should be noted that an essential difference between the two controller structures is that the controllers based on state feedback incorporate some form of model of the system in order to estimate the states, and this state information gives this type of controller much greater potential for "tuning" the control action. In particular, it is feasible to alter the response of certain states of the system without grossly affecting the remainder of the system (within the usual controllability and observability constraints, and in the case of flexible structures, spillover constraints). Obviously, the larger the number of states of the system that are reconstructed, then the greater the potential for control "tuning", but this has to be offset against the increased computational cost of estimating a large number of states. Controllers based on output feedback (where the states are not estimated), generally have less potential for this "tuning" of the control action for two reasons. Firstly, there are generally fewer parameters in an output feedback matrix, as opposed to a state feedback matrix, offering less facility for adjustments, and secondly, the alteration of a single parameter of an output feedback matrix generally results in the alteration of more than one state of the system, thus making it difficult to make slight alterations to the characteristics of the closed loop system. Thus, to some extent, the choice of controller type (that is based on state feedback or output feedback) may be dictated by the control system performance specifications.

For example, consider the situation of having designed a controller for a flexible spacecraft based on output feedback which produces an acceptable rigid mode response time, but the responses contain elastic mode components which have unacceptable characteristics, such as insufficient damping

and long time constants. The output feedback controller can be redesigned with higher gains which will increase the damping and reduce the time constants of the elastic modes, but it will also increase the damping of the rigid modes and reduce their time constants. This overall increase in system response time will require higher levels of control effort, something which is generally desirable to avoid in spacecraft system design due to their limited energy resources. However, if the controller was based on state feedback, and assuming that the states corresponding to the offending mode or modes were available from the estimator, then the gain terms corresponding to these states could be redesigned giving the required improvement in the responses without having to alter the rigid mode responses and consequently incurring only minor additional control effort requirements.

8.6 Summary

In this chapter the techniques introduced in earlier chapters for the design of controllers for flexible structures have been applied to a simple example, the experimental beam described in Chapter 6.

The derivation of a nominal model for control system design and two perturbed models for evaluating the performance of the resulting designs has been described. The first perturbed model is essentially similar to the nominal model, the structure still having a fairly uniform mass distribution but higher overall mass, which results in lowering of the modal frequencies, but little change in the modeshapes. The second perturbed model has a non-uniform mass distribution as well as increased overall mass, which results in significant changes in the modeshapes as well as shifting of the modal frequencies.

The computational problems involved with the design of controllers for flexible structures have been highlighted, and the suitability of these methods for general application to large flexible spacecraft control problems has consequently been inferred.

The results of simulation studies of the various controllers designed for the experimental beam have been presented and compared, again adding conviction to the suitability of these design methods to more general applications. This refinement of suitable design methods for general application to more realistic and complex problems leads to the next chapter, where such an example is considered.

Chapter 9

Controller Design for a Large Flexible Spacecraft

9.1 Introduction

In this chapter the design of a controller for a realistic example of a large flexible spacecraft is considered. The application to the experimental beam, as discussed in the previous chapter, identified those methods suitable for such a complex problem.

The spacecraft considered is based on an early version (November 1985) of the European Space Agency (ESA) Space Platform, which forms part of the Columbus program. The details of the spacecraft and the derivation of a mathematical model are described in the next section.

The following section describes the design of a controller for all six degrees of freedom of the overall spacecraft. As will be explained later, this representation of the position of a spacecraft in free space is rather naive, and thus the subsequent section considers the more usual three axis attitude control problem. The performance requirement for the attitude controller is deliberately set quite high to emphasise the point that "spillover" problems (see Chapter 4) are generally compounded as the controller performance requirements increase.

Aspects of the two controller design exercises are compared in the next section, with a view to identifying trends for better understanding of the modelling/control system design problem.

9.2 The Space Platform

The spacecraft which forms the subject of attention of this chapter is an early version of the ESA Space Platform. In particular, the version considered is the initial operational configuration of the Polar mission. The details of this version are based on information available at November 1985, which was an early stage in the development of the Space Platform, and thus the flight version may be very different from the spacecraft portrayed here. A simple sketch of the spacecraft is shown in figure 9.1, where the major subassemblies can be identified.

The propulsion module is a self-contained unit which provides the orbit manoeuvring thrusts, the utilities module contains the essential "housekeeping" subsystems, such as the attitude and orbit control system (AOCS), the power distribution system, *etc.* The berthing module supports the various payloads that the spacecraft will carry and provides a means of docking with other spacecraft, and the solar arrays provide electrical power for the spacecraft.

The spacecraft is too large to be launched via a single flight of a launch vehicle, so it is intended to launch the spacecraft in two parts and assemble the spacecraft in orbit. This approach of in-orbit construction is likely to be a common feature of future large spacecraft projects.

The spacecraft is represented using a finite element model, in a similar manner to that described for the experimental beam discussed in the previous chapter, the representation being shown in figure 9.2. The solar arrays, the heat radiator, and the spacer tubes were modelled with distributed masses, but the utilities module, propulsion module, berthing module, and payloads were modelled with lumped masses, because these latter elements have as yet unknown mass distributions. The various modules of the structure are connected by berthing units, which were modelled as a set of six springs linking the six degrees of freedom of each module independently. The utilities module and the berthing module were modelled as lumped masses at their centres of gravity, with massless beams between the centres of gravity and the connection points to other parts of the structure. The stiffnesses of these massless beams were set to be similar to the stiffness of a cuboid shell of appropriate size. The propulsion module was similarly modelled, except that the stiffness of the massless beam was calculated from an equivalent solid section, as opposed to a shell section, due to the higher mass density of the propulsion module. The spacers are 1.1 metre diameter tubes with a wall thickness of 2.5 mm and are constructed of carbon fibre material.

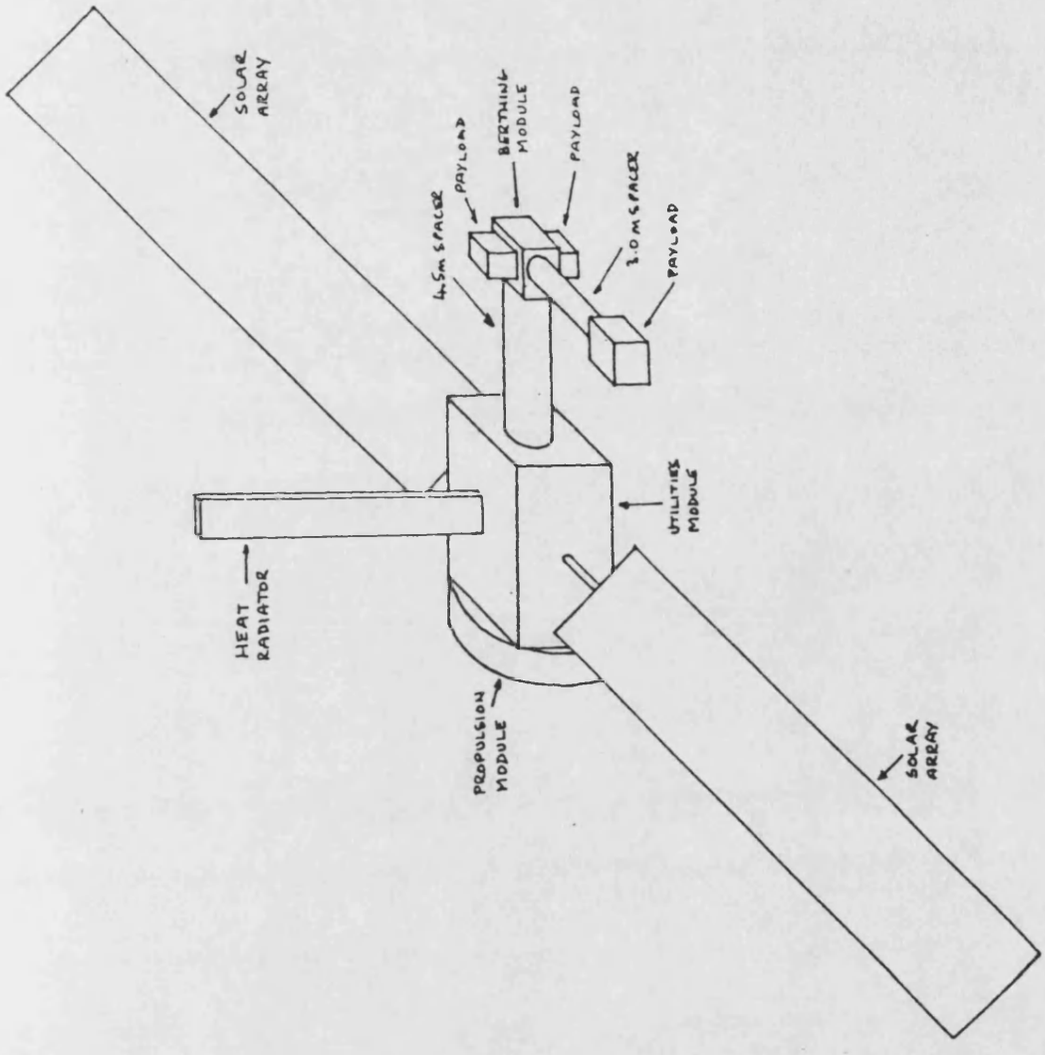


Figure 9.1: Sketch of Space Platform

They were modelled with beams of appropriate sectional properties with a mass density corresponding to the carbon fibre material. The payloads were modelled as lumped masses at the end of massless beams with stiffnesses set to produce fundamental cantilever frequencies of 3Hz. in torsion and bending modes. This was done in order to comply in a "worst case" manner with a specification for Platform payloads which requires that all cantilever frequencies of any payload to be greater than 3Hz.

Only limited details of the solar arrays and heat radiator were available, so data was obtained by extrapolating data from solar arrays used on large satellites such as the B.Ae. Olympus. The length of the solar arrays on the Space Platform was taken to be 20 metres. The heat radiator was assumed to have a length of 9 metres. The solar arrays and heat radiator were modelled by simple beams with appropriate sectional properties, and with distributed mass corresponding to the array and radiator masses. Details of the elements are listed in tables 9.1 and 9.2.

The structural analysis program NASTRAN [32], [33] was used to compute the vibration frequencies (modal frequencies) and the modeshape data. The frequencies below 10Hz. are listed in table 9.3, the first six values are the zero frequency rigid modes of the structure. The mode shape matrix for node 1 is shown in table 9.4.

The state space model of the spacecraft was again derived in the manner described in Chapter 2. Note that no actuator or sensor dynamics have been included because when this model was derived no information was available regarding actuators and sensors. However, it is known that the actuators and sensors are most likely to be located in the utilities module, which corresponds to node 1 of the finite element model, thus the actuators and sensors are assumed to act at this point. It has been assumed that translational forces can be generated in all three directions, and that rotational torques can be generated around all three axes. The resulting state space model is referred to as the nominal platform model and is 52nd order.

A perturbed model was derived by altering the modulus of elasticity of the material from 1.1×10^{11} to 1.0×10^{10}. As would be expected, this caused a small reduction in the modal frequencies, but negligible change in the modeshapes. The frequencies of the perturbed model are shown in table 9.5, and the modeshape matrix for node 1 is shown in table 9.6. This model is referred to as the perturbed platform model and is also 52nd order.

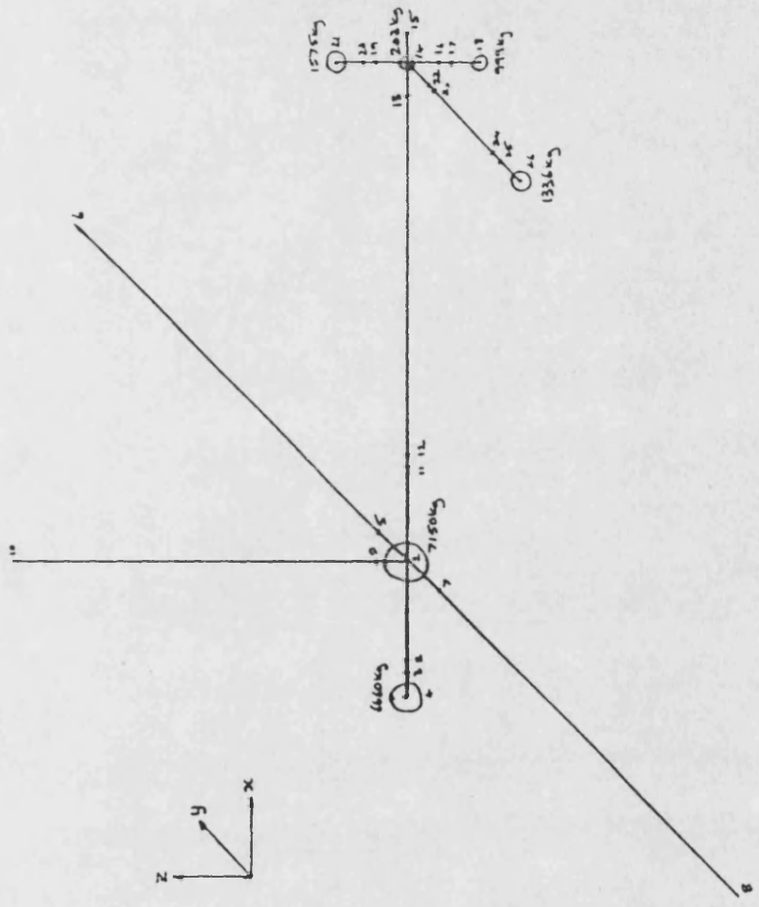


Figure 9.2: Finite element representation of the Space Platform

Beam No.	Grid Points		L (m)	A (m^2)	I1 (m^4)	I2 (m^4)	J (m^4)	E ($N.m^{-2}$)	Density ($Kg.m^{-3}$)
12	1	2	2.48	19.175	33.7	33.7	33.7	1.1×10^{11}	0
15	1	5	1.654	19.175	33.7	33.7	33.7	1.1×10^{11}	0
17	1	7	1.596	19.175	33.7	33.7	33.7	1.1×10^{11}	0
19	1	9	0.809	19.175	33.7	33.7	33.7	1.1×10^{11}	0
111	1	11	3.42	19.175	33.7	33.7	33.7	1.1×10^{11}	0
34	3	4	0.48	7.0685	5.422	5.422	7.496	1.1×10^{11}	0
56	5	6	20.0	0.2	4.17×10^{-3}	2.667×10^{-3}	2.667×10^{-4}	1.1×10^{11}	200.0
78	7	8	20.0	0.2	4.17×10^{-3}	2.667×10^{-3}	2.667×10^{-4}	1.1×10^{11}	200.0
910	9	10	9.0	0.1	2.08×10^{-3}	2.084×10^{-3}	2.667×10^{-3}	1.1×10^{11}	600.0
1213	12	13	4.5	8.62×10^{-3}	1.238×10^{-3}	1.238×10^{-3}	2.565×10^{-3}	1.1×10^{11}	1600.0
1314	13	14	0.75	2.25	16.85	16.85	16.85	1.1×10^{11}	0
1415	14	15	0.75	2.25	16.85	16.85	16.85	1.1×10^{11}	0
1416	14	16	0.75	2.25	16.85	16.85	16.85	1.1×10^{11}	0
1419	14	19	0.75	2.25	16.85	16.85	16.85	1.1×10^{11}	0
1422	14	22	0.75	2.25	16.85	16.85	16.85	1.1×10^{11}	0
1718	17	18	0.7	0.75	9.14×10^{-3}	9.14×10^{-3}	7.34×10^{-4}	1.1×10^{11}	0
2021	20	21	0.7	0.75	1.44×10^{-7}	1.44×10^{-7}	2.36×10^{-3}	1.1×10^{11}	0
2324	23	24	3.0	8.62×10^{-3}	1.238×10^{-3}	1.238×10^{-3}	2.595×10^{-3}	1.1×10^{11}	1600.0
2526	25	26	0.7	0.75	1.22×10^{-7}	1.22×10^{-7}	2.0×10^{-3}	1.1×10^{11}	0
23	2	3	0.3	Six independent springs with spring constant $K = 1\times10^{7}\ N.m^{-1}$					
1112	11	12	0.3						
1617	16	17	0.3						
1920	19	20	0.3						
2223	22	23	0.3						
2425	24	25	0.3						

Table 9.1: Properties of beam elements for Platform model

Grid point	Mass (kg)
1	7150.0
4	6660.0
14	203.0
18	999.0
21	1575.0
26	1336.0

Table 9.2: Properties of mass elements

Mode no.	Frequency (Hz)	Mode no.	Frequency (Hz)
1	0	14	1.5342
2	0	15	1.6169
3	0	16	2.7599
4	0	17	3.3073
5	0	18	3.9500
6	0	19	4.2180
7	0.4874	20	4.6930
8	0.5347	21	4.7336
9	1.2466	22	4.9241
10	1.2783	23	5.9263
11	1.3988	24	6.2460
12	1.4723	25	8.9136
13	1.5055	26	9.9334

Table 9.3: Modal frequencies of nominal Platform

-0.6613E-05	-0.1951E-03	0.1354E-02	0.6504E-03	0.1517E-03	0.7528E-03
-0.2837E-03	-0.1025E-04	-0.5995E-02	0.1784E-03	-0.8372E-03	0.1407E-03
0.2366E-04	0.4744E-04	0.2753E-03	0.1007E-02	-0.8519E-05	-0.5422E-03
-0.7281E-03	-0.5600E-05	0.3607E-02	0.5472E-04	-0.1036E-02	-0.5718E-05
-0.5474E-02	0.4396E-02	-0.9338E-04	-0.4257E-05	0.1037E-03	-0.2922E-04
0.4367E-02	0.5552E-02	0.1096E-03	0.1171E-04	-0.7994E-04	-0.7730E-04
0.1523E-02	-0.1966E-04	0.2670E-05	-0.6111E-06	0.1555E-05	0.3084E-05
0.2508E-03	0.6130E-03	0.5032E-06	0.2021E-04	-0.8163E-05	-0.4552E-03
-0.1189E-02	-0.1505E-02	0.5305E-03	-0.6145E-04	-0.9475E-04	-0.5427E-03
0.4386E-03	0.6731E-03	0.1690E-02	-0.1770E-03	-0.5081E-03	0.2069E-03
-0.3282E-03	0.1542E-03	0.3005E-03	-0.1241E-03	-0.2605E-03	0.2517E-04
-0.3635E-03	0.1059E-04	0.8956E-04	-0.1593E-04	-0.2548E-03	-0.8458E-06
0.4655E-03	-0.1401E-03	0.1084E-02	0.3039E-04	0.5851E-03	-0.2839E-04
0.7329E-03	0.2671E-02	-0.2036E-03	0.4913E-04	0.1010E-03	0.5424E-03
-0.3297E-02	0.1025E-02	-0.1679E-04	0.2605E-04	0.5836E-04	0.1718E-03
0.7926E-04	0.4924E-04	0.3618E-02	0.1139E-04	-0.1455E-02	0.1395E-04
0.2361E-03	-0.1658E-02	0.6506E-04	-0.1922E-04	-0.5118E-04	-0.3947E-03
-0.2174E-03	0.1380E-03	0.2314E-02	0.1870E-05	-0.4553E-03	-0.6389E-05
-0.1411E-04	-0.3097E-02	0.3504E-03	-0.1031E-03	0.1335E-04	0.1434E-03
-0.8571E-04	0.4552E-04	0.3169E-03	-0.1149E-04	0.4828E-04	0.4601E-06
0.4969E-04	0.3107E-03	0.2872E-04	-0.1022E-03	0.1344E-05	0.1935E-03
0.8385E-04	0.1062E-02	0.4524E-04	-0.9791E-03	-0.2374E-04	-0.1126E-03
0.5506E-02	0.7382E-03	0.2574E-02	0.8632E-04	0.4870E-03	-0.1007E-03
-0.1637E-02	0.2366E-02	-0.9427E-03	0.1961E-03	-0.1802E-03	-0.2654E-03
-0.3844E-02	0.6170E-03	0.5712E-02	0.3967E-04	0.8887E-03	-0.2498E-04
-0.3279E-03	-0.4919E-02	0.5292E-03	-0.2086E-03	0.6930E-04	0.2054E-03

Table 9.4: Modeshape matrix for node 1 of nominal Platform

Mode no.	Frequency (Hz)	Mode no.	Frequency (Hz)
1	0	14	1.4636
2	0	15	1.5338
3	0	16	2.7272
4	0	17	3.2224
5	0	18	3.7744
6	0	19	4.1980
7	0.4647	20	4.4810
8	0.5102	21	4.5149
9	1.2015	22	4.7183
10	1.2283	23	5.9134
11	1.3330	24	6.2420
12	1.3937	25	8.9078
13	1.4301	26	9.9269

Table 9.5: Modal frequencies of perturbed Platform

-0.1966E-03	-0.4244E-03	-0.1941E-02	0.6766E-03	-0.1839E-03	0.7145E-03
0.2933E-03	0.8118E-03	0.4607E-02	-0.5071E-03	0.4610E-03	0.5370E-03
0.3748E-03	-0.1262E-02	0.4296E-02	0.8226E-03	0.4800E-03	-0.2433E-03
-0.7359E-03	-0.1020E-03	0.2713E-02	0.1342E-03	-0.1145E-02	-0.6928E-05
0.2784E-03	0.6906E-02	0.1561E-03	0.2485E-03	-0.6991E-06	-0.1703E-03
0.6983E-02	-0.2371E-03	-0.1903E-03	0.9261E-06	-0.1605E-03	-0.1643E-04
0.1524E-02	-0.1814E-04	0.2632E-05	-0.5802E-06	0.1615E-05	0.3023E-05
-0.2460E-03	-0.6028E-03	-0.4905E-06	-0.2009E-04	0.8032E-05	0.4582E-03
0.1130E-02	0.1329E-02	-0.6101E-03	0.6998E-04	0.1140E-03	0.5003E-03
0.5121E-03	0.7089E-03	0.1583E-02	-0.1701E-03	-0.4660E-03	0.2285E-03
-0.2792E-03	0.1665E-03	0.2991E-03	-0.1251E-03	-0.2447E-03	0.2912E-04
0.3069E-03	-0.1949E-04	0.2063E-06	0.1649E-04	0.2122E-03	-0.1024E-05
-0.4434E-03	0.2287E-03	0.1102E-02	-0.4146E-04	-0.6096E-03	0.4741E-04
0.9738E-03	0.2650E-02	-0.2251E-03	0.4885E-04	0.1125E-03	0.5486E-03
-0.3278E-02	0.1147E-02	-0.3467E-04	0.2798E-04	0.6065E-04	0.1960E-03
0.6657E-04	0.5147E-04	0.3592E-02	0.9725E-05	-0.1458E-02	0.1440E-04
-0.2305E-03	0.1750E-02	-0.6584E-04	0.1746E-04	0.5037E-04	0.4108E-03
-0.1789E-03	0.7193E-04	0.2448E-02	0.1387E-05	-0.4758E-03	-0.3834E-05
0.6137E-05	0.2990E-02	-0.3064E-03	0.2009E-03	-0.1963E-04	-0.1288E-03
-0.2730E-03	0.3444E-04	0.2437E-03	-0.9218E-05	0.3635E-04	0.1195E-05
-0.4311E-04	-0.2422E-03	-0.3416E-04	0.1179E-03	-0.1595E-05	-0.2005E-03
-0.6370E-04	-0.1308E-02	-0.7816E-05	0.9662E-03	0.2866E-04	0.1254E-03
0.5505E-02	0.7331E-03	0.2570E-02	0.7628E-04	0.4914E-03	-0.9865E-04
-0.1595E-02	0.2381E-02	-0.9198E-03	0.1818E-03	-0.1778E-03	-0.2660E-03
-0.3834E-02	0.6157E-03	0.5704E-02	0.3929E-04	0.8899E-03	-0.2516E-04
0.3294E-03	0.4882E-02	-0.5299E-03	0.2114E-03	-0.6959E-04	-0.2050E-03

Table 9.6: Modeshape matrix for node 1 of perturbed Platform

9.3 Six Degree of Freedom Controller

The requirements considered for the design of a controller for the platform model is that the response time of the rigid modes is not particularly important, but robustness of the closed loop system is important. It has already been noted (see Chapter 4) that if the controller bandwidth is designed to be lower than the frequencies of the elastic modes, then stability is generally easier to achieve, and also good robustness is generally easier to achieve. Thus the response time of the closed loop system has deliberately been designed to be relatively slow.

From the experience derived with the application to the experimental beam (see Chapter 8), the range of methods available for the design of the controller is much reduced, due primarily to the unsuitability of most of the methods to dealing with high order problems.

Similar design philosophies to those presented in the previous chapter have been followed, that is the two state feedback based approaches of model order reduction – controller design, and controller design – controller order reduction, and the output feedback based approach where the dynamics of the elastic modes are ignored.

9.3.1 Baseline Controller

In order to be able to compare the performance of the various controllers, a reference controller was designed for the nominal platform model, based on full state feedback using a 52nd order state estimator. The state feedback matrix and the state estimator gain matrix were computed using the program LQGDES (see Chapter 7), using unit weighting matrices. The eigenvalues of the resulting closed loop system are shown in table 9.7. The closed loop system formed by the nominal platform model with this 52nd order controller was simulated using the experimental rig digital computing system, as described in Chapters 6 and 7. The responses to a unit step demand for the first translational position input is shown in figure 9.3, the response to a unit step demand on the first rotational position input is shown in figure 9.4, and the response to a unit step demand on all six inputs is shown in figure 9.5. Note that in the figures the outputs 1, 3, and 5 correspond to translational position of the spacecraft, and outputs 7, 9, and 11 correspond to rotational position of the spacecraft.

It can be seen from the responses that the closed loop system is completely stable, but it is interesting to observe the interaction between the six positional inputs and outputs. Note also that the responses show negligible signs of high frequency dynamics because the elastic modes are not excited by this narrow bandwidth controller.

The theory in Chapter 4 suggests that such a closed loop system should be relatively insensitive to parameter variations associated with the elastic modes. To examine this, the closed loop system formed by this 52nd order controller and the perturbed platform model was simulated. The same three step responses cases were examined and the results are shown in figures 9.6, 9.7, and 9.8.

It can be seen that there is negligible difference between the responses of the nominal model and the responses of the perturbed model, which tends to uphold the theory.

Although the performance of this 52nd order controller is acceptable, obviously it would be desirable to find a controller which offers similar performance but which is of lower order. The first method considered is the model order reduction – controller design approach.

Designed Eigenvalues Of Plant		Designed Eigenvalues Of Estimator	
Real	Imaginary	Real	Imaginary
-0.31206	62.412	-0.34845	62.413
-0.31206	-62.412	-0.34845	-62.413
-0.28007	56.006	-0.34139	56.006
-0.28007	-56.006	-0.34139	-56.006
-0.19621	39.245	-0.20513	39.245
-0.19621	-39.245	-0.20513	-39.245
-0.18623	37.235	-0.21856	37.235
-0.18623	-37.235	-0.21856	-37.235
-0.15470	30.939	-0.15633	30.939
-0.15470	-30.939	-0.15633	-30.939
-0.14870	29.742	-0.14753	29.486
-0.14870	-29.742	-0.14753	-29.486
-0.14745	29.486	-0.14881	29.742
-0.14745	-29.486	-0.14881	-29.742
-0.13251	26.502	-0.13882	26.502
-0.13251	-26.502	-0.13882	-26.502
-0.12411	24.818	-0.12755	24.818
-0.12411	-24.818	-0.12755	-24.818
-0.10390	20.780	-0.10544	20.780
-0.10390	-20.780	-0.10544	-20.780
-0.86722E-01	17.341	-0.93103E-01	17.341
-0.86722E-01	-17.341	-0.93103E-01	-17.341
-0.50830E-01	10.159	-0.53806E-01	10.159
-0.50830E-01	-10.159	-0.53806E-01	-10.159
-0.39164E-01	7.8328	-0.40004E-01	7.8328
-0.39164E-01	-7.8328	-0.40004E-01	-7.8328
-0.40162E-01	8.0316	-0.40936E-01	8.0317
-0.40162E-01	-8.0316	-0.40936E-01	-8.0317
-0.43951E-01	8.7887	-0.44019E-01	8.7887
-0.43951E-01	-8.7887	-0.44019E-01	-8.7887
-0.48221E-01	9.6394	-0.50135E-01	9.6394
-0.48221E-01	-9.6394	-0.50135E-01	-9.6394
-0.47305E-01	9.4591	-0.47723E-01	9.4591
-0.47305E-01	-9.4591	-0.47723E-01	-9.4591
-0.46251E-01	9.2507	-0.46299E-01	9.2507
-0.46251E-01	-9.2507	-0.46299E-01	-9.2507
-0.16805E-01	3.3597	-0.16864E-01	3.3597
-0.16805E-01	-3.3597	-0.16864E-01	-3.3597
-0.15321E-01	3.0625	-0.15516E-01	3.0625
-0.15321E-01	-3.0625	-0.15516E-01	-3.0625
-0.21624E-01	0.21613E-01	-0.21629E-01	0.21608E-01
-0.21624E-01	-0.21613E-01	-0.21629E-01	-0.21608E-01
-0.24439E-01	0.24425E-01	-0.24447E-01	0.24418E-01
-0.24439E-01	-0.24425E-01	-0.24447E-01	-0.24418E-01
-0.25966E-01	0.25949E-01	-0.25975E-01	0.25940E-01
-0.25966E-01	-0.25949E-01	-0.25975E-01	-0.25940E-01
-0.59839E-01	0.59626E-01	-0.59946E-01	0.59520E-01
-0.59839E-01	-0.59626E-01	-0.59946E-01	-0.59520E-01
-0.59645E-01	0.59434E-01	-0.59751E-01	0.59328E-01
-0.59645E-01	-0.59434E-01	-0.59751E-01	-0.59328E-01
-0.59442E-01	0.59233E-01	-0.59547E-01	0.59129E-01
-0.59442E-01	-0.59233E-01	-0.59547E-01	-0.59129E-01

Table 9.7: Closed loop poles of Platform with baseline controller

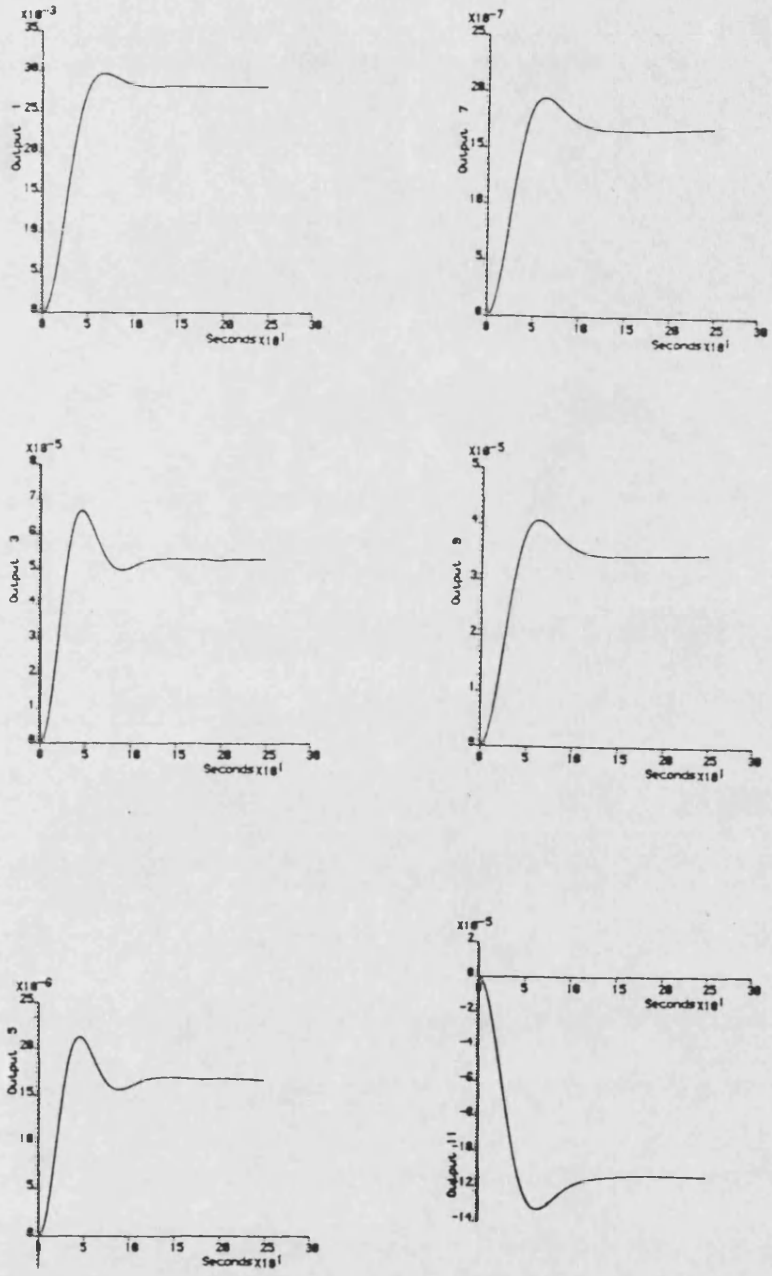


Figure 9.3: Step responses of input 1 of nominal Platform with baseline controller

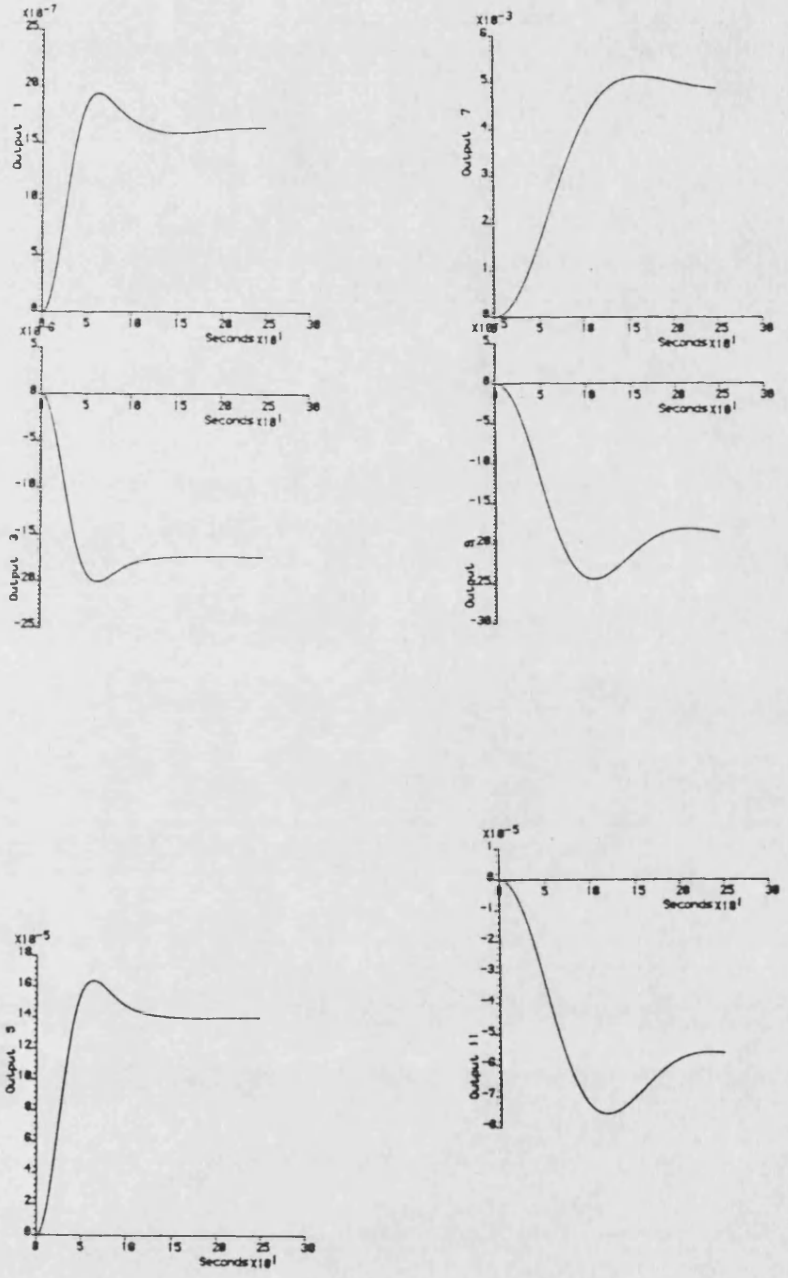


Figure 9.4: Step responses of input 4 of nominal Platform with baseline controller

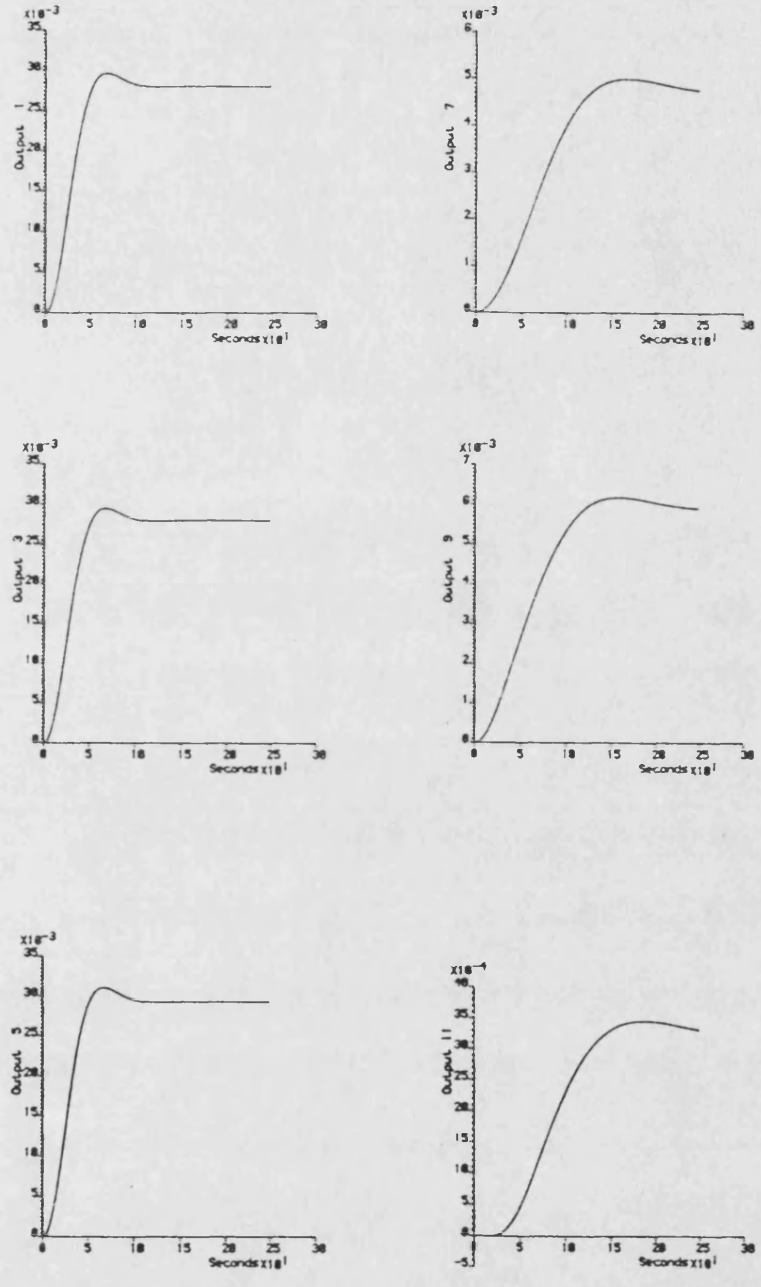


Figure 9.5: Step responses of all inputs of nominal Platform with baseline controller

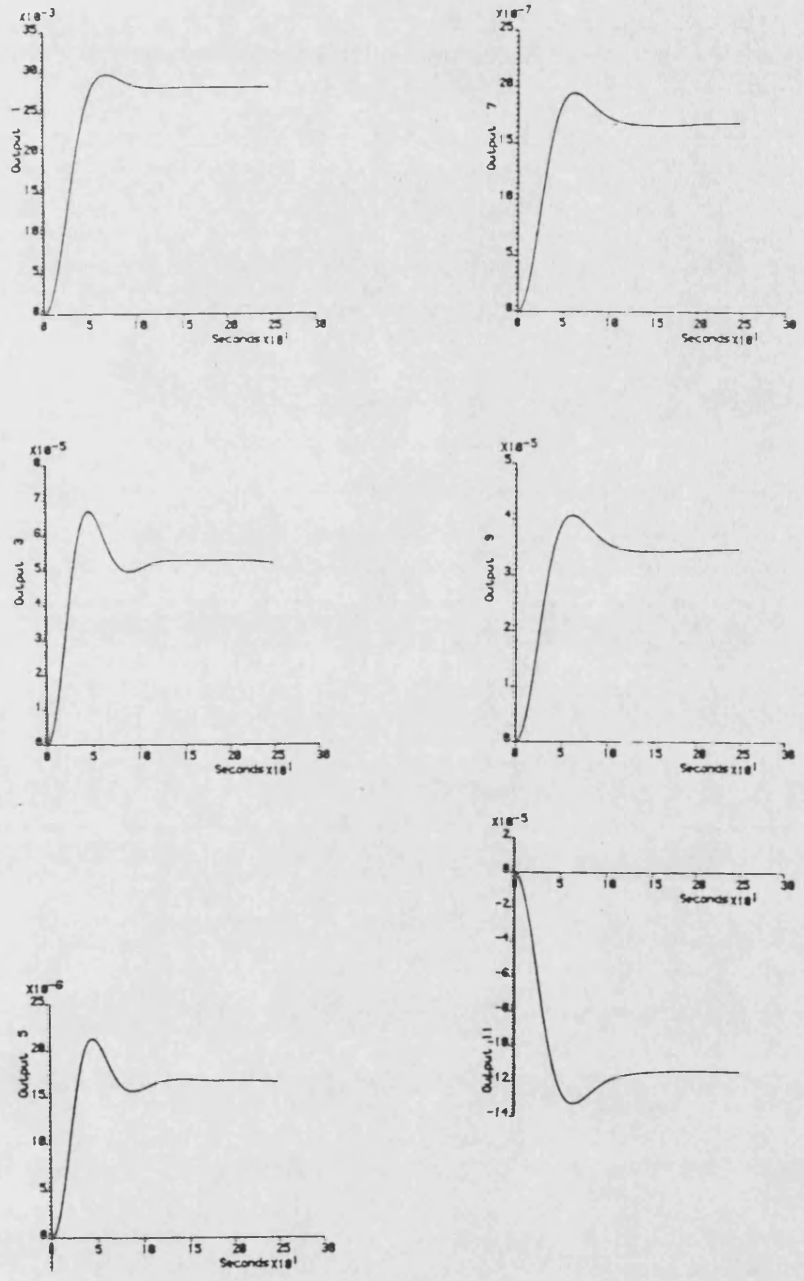


Figure 9.6: Step responses of input 1 of perturbed Platform with baseline controller

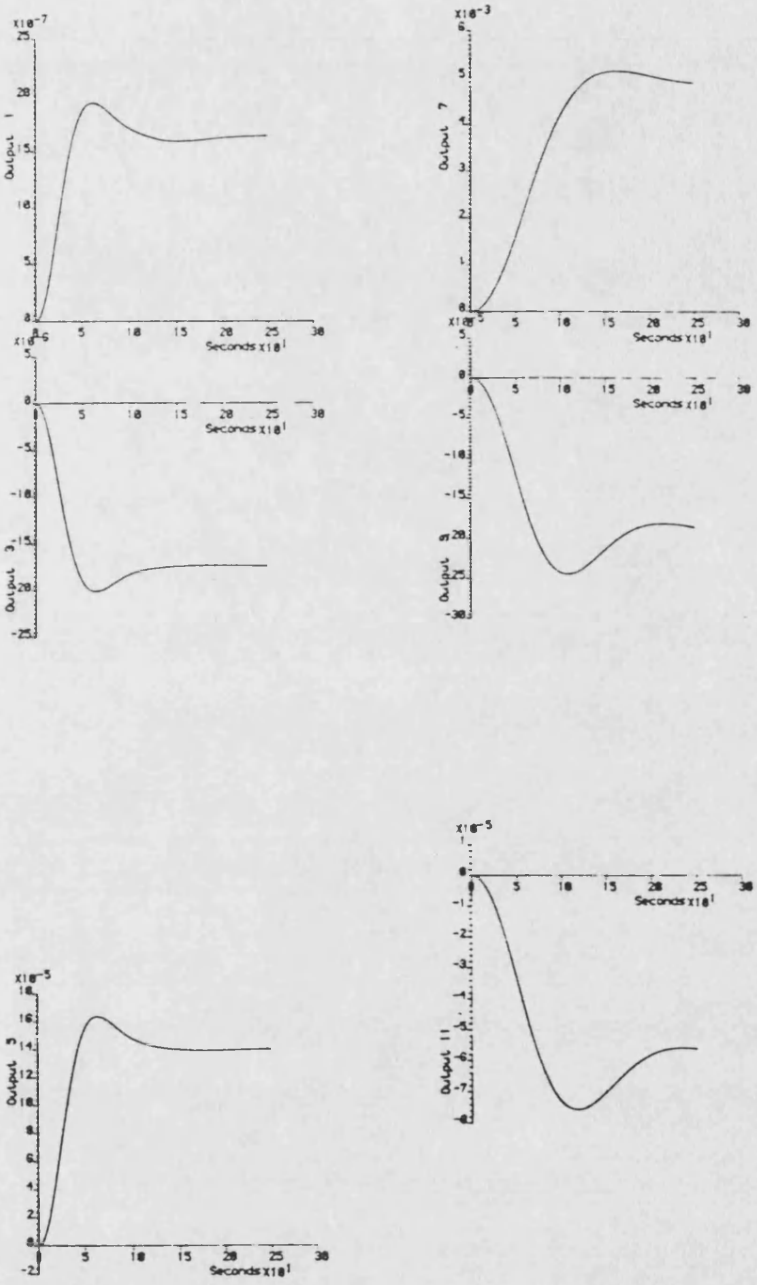


Figure 9.7: Step responses of input 4 of perturbed Platform with baseline controller

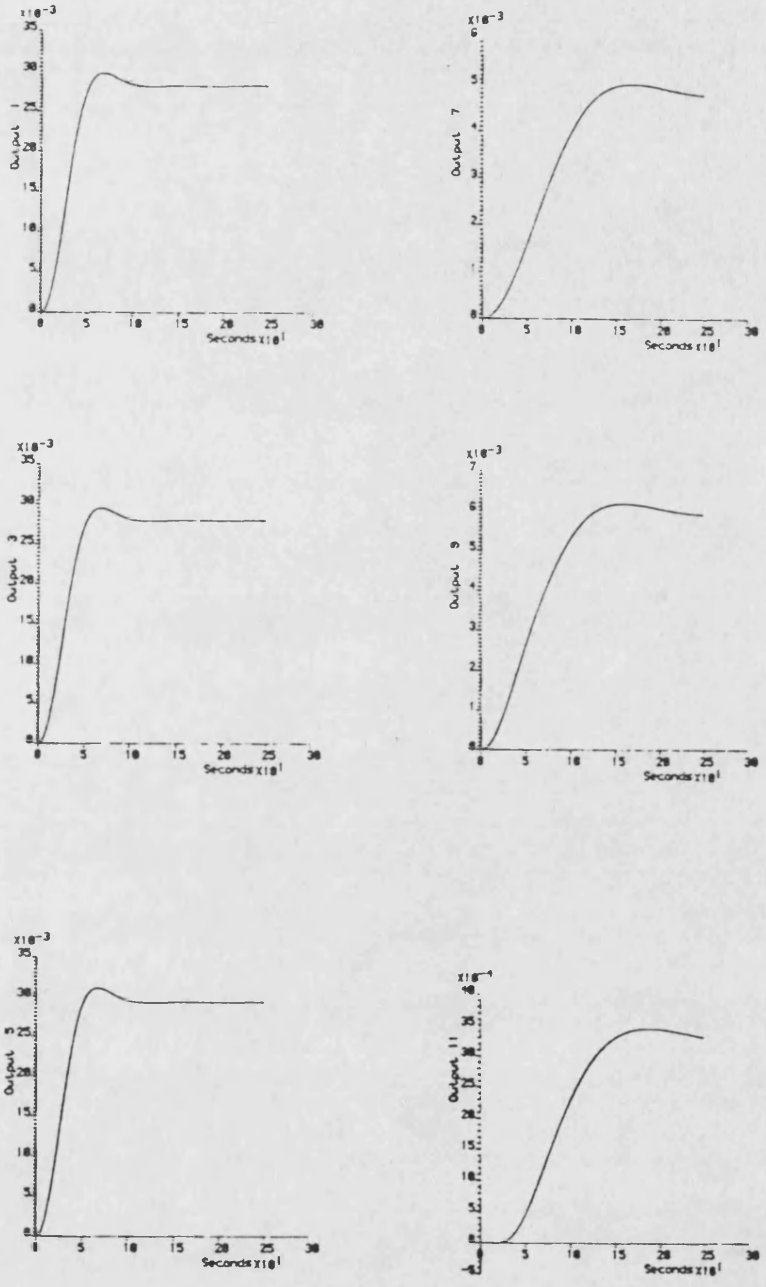


Figure 9.8: Step responses of all inputs of perturbed Platform with baseline controller

9.3.2 Model Order Reduction—Controller Design

In the same manner as was used for the experimental beam controller, the order of the design model was reduced via modal cost analysis and mode deletion using the program MCA. The results of the modal cost analysis are shown in table 9.8 and figure 9.9.

Mode no.	Modal Cost	Mode no.	Modal Cost
15	0.16638×10^{-3}	24	0.33536×10^{-4}
23	0.14333×10^{-3}	18	0.32069×10^{-4}
16	0.12408×10^{-3}	8	0.27203×10^{-4}
25	0.12265×10^{-3}	13	0.26249×10^{-4}
14	0.11761×10^{-3}	17	0.20191×10^{-4}
7	0.10720×10^{-3}	22	0.96370×10^{-5}
9	0.77096×10^{-4}	11	0.49177×10^{-5}
10	0.67542×10^{-4}	12	0.31403×10^{-5}
26	0.55917×10^{-4}	21	0.70256×10^{-6}
19	0.51996×10^{-4}	20	0.53849×10^{-6}

Table 9.8: Modal cost analysis of Platform model

The first reduced order model derived was based on the rigid modes plus the first seven elastic modes with the highest cost. The resulting 26th order model has a model error of about 28%.

The state feedback and state estimator gain matrices were computed using the program LQGDES using unit weighting matrices, and the poles of the complete closed loop system are shown in table 9.9, which shows the system to be stable. This system was simulated, and the three step response cases as used above are shown in figures 9.10, 9.11, and 9.12.

It can be seen that no noticeable degradation in performance has occurred with the use of this controller over the full 52nd order controller. The simulation results for this 26th order controller on the perturbed model are shown in figures 9.13, 9.14, and 9.15.

Subsequently, further reductions in the order of the model were considered. In particular, a model was formed from the rigid modes and the first three elastic modes with the highest cost, which resulted in an 18th order model with a model error of about 64%. Again a controller was designed using the program LQGDES with unit weighting matrices, and the closed

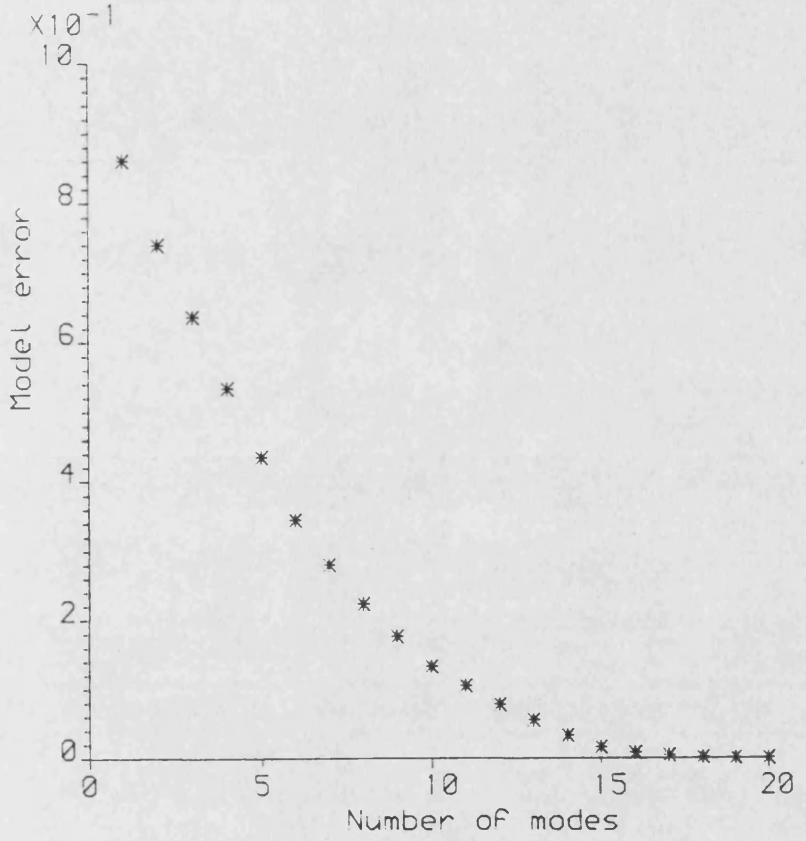


Figure 9.9: Model error function for Platform model

Designed Eigenvalues Of Plant	
Real	Imaginary
-0.28007	56.006
-0.28007	-56.006
-0.18623	37.235
-0.18623	-37.235
-0.86722E-01	17.341
-0.86722E-01	-17.341
-0.50830E-01	10.159
-0.50830E-01	-10.159
-0.48221E-01	9.6394
-0.48221E-01	-9.6394
-0.39164E-01	7.8328
-0.39164E-01	-7.8328
-0.15321E-01	3.0625
-0.15321E-01	-3.0625
-0.21624E-01	0.21613E-01
-0.21624E-01	-0.21613E-01
-0.24439E-01	0.24425E-01
-0.24439E-01	-0.24425E-01
-0.25966E-01	0.25949E-01
-0.25966E-01	-0.25949E-01
-0.59839E-01	0.59626E-01
-0.59839E-01	-0.59626E-01
-0.59645E-01	0.59434E-01
-0.59645E-01	-0.59434E-01
-0.59442E-01	0.59233E-01
-0.59442E-01	-0.59233E-01

Designed Eigenvalues Of Estimator	
Real	Imaginary
-0.34139	56.006
-0.34139	-56.006
-0.21857	37.235
-0.21857	-37.235
-0.93103E-01	17.341
-0.93103E-01	-17.341
-0.53806E-01	10.159
-0.53806E-01	-10.159
-0.50135E-01	9.6394
-0.50135E-01	-9.6394
-0.40004E-01	7.8328
-0.40004E-01	-7.8328
-0.15516E-01	3.0625
-0.15516E-01	-3.0625
-0.59946E-01	0.59520E-01
-0.59946E-01	-0.59520E-01
-0.59751E-01	0.59328E-01
-0.59751E-01	-0.59328E-01
-0.59547E-01	0.59129E-01
-0.59547E-01	-0.59129E-01
-0.25975E-01	0.25940E-01
-0.25975E-01	-0.25940E-01
-0.24447E-01	0.24418E-01
-0.24447E-01	-0.24418E-01
-0.21629E-01	0.21608E-01
-0.21629E-01	-0.21608E-01

Eigenvalues Of Closed Loop Syste	
Real	Imaginary
-0.31205	62.412
-0.31205	-62.412
-0.28007	56.006
-0.28007	-56.006
-0.34139	56.006
-0.34139	-56.006
-0.19620	39.245
-0.19620	-39.245
-0.18623	37.235
-0.18623	-37.235
-0.21857	37.235
-0.21857	-37.235
-0.15470	30.939
-0.15470	-30.939
-0.14870	29.742
-0.14870	-29.742
-0.14745	29.486
-0.14745	-29.486
-0.13250	26.502
-0.13250	-26.502
-0.12410	24.818
-0.12410	-24.818
-0.10390	20.780
-0.10390	-20.780
-0.86722E-01	17.341
-0.86722E-01	-17.341
-0.93103E-01	17.341
-0.93103E-01	-17.341
-0.50830E-01	10.159
-0.50830E-01	-10.159
-0.53806E-01	10.159
-0.53806E-01	-10.159
-0.40149E-01	8.0316
-0.40149E-01	-8.0316
-0.43950E-01	8.7887
-0.43950E-01	-8.7887
-0.47300E-01	9.4591
-0.47300E-01	-9.4591
-0.46250E-01	9.2507
-0.46250E-01	-9.2507
-0.48221E-01	9.6394
-0.48221E-01	-9.6394
-0.50135E-01	9.6394
-0.50135E-01	-9.6394
-0.39164E-01	7.8328
-0.39164E-01	-7.8328
-0.40004E-01	7.8328
-0.40004E-01	-7.8328
-0.16799E-01	3.3597
-0.16799E-01	-3.3597
-0.15321E-01	3.0625
-0.15321E-01	-3.0625
-0.15516E-01	3.0625
-0.15516E-01	-3.0625
-0.21486E-01	0.21644E-01
-0.21486E-01	-0.21644E-01
-0.21767E-01	0.21577E-01
-0.21767E-01	-0.21577E-01
-0.26062E-01	0.25922E-01
-0.26062E-01	-0.25922E-01
-0.25879E-01	0.25966E-01
-0.25879E-01	-0.25966E-01
-0.24374E-01	0.24438E-01
-0.24374E-01	-0.24438E-01
-0.24513E-01	0.24404E-01
-0.24513E-01	-0.24404E-01
-0.60166E-01	0.59494E-01
-0.60166E-01	-0.59494E-01
-0.59979E-01	0.59306E-01
-0.59979E-01	-0.59306E-01
-0.59625E-01	0.59645E-01
-0.59625E-01	-0.59645E-01
-0.59612E-01	0.59131E-01
-0.59612E-01	-0.59131E-01
-0.59414E-01	0.59460E-01
-0.59414E-01	-0.59460E-01
-0.59375E-01	0.59231E-01
-0.59375E-01	-0.59231E-01

Table 9.9: Closed loop poles of platform with 26th order controller

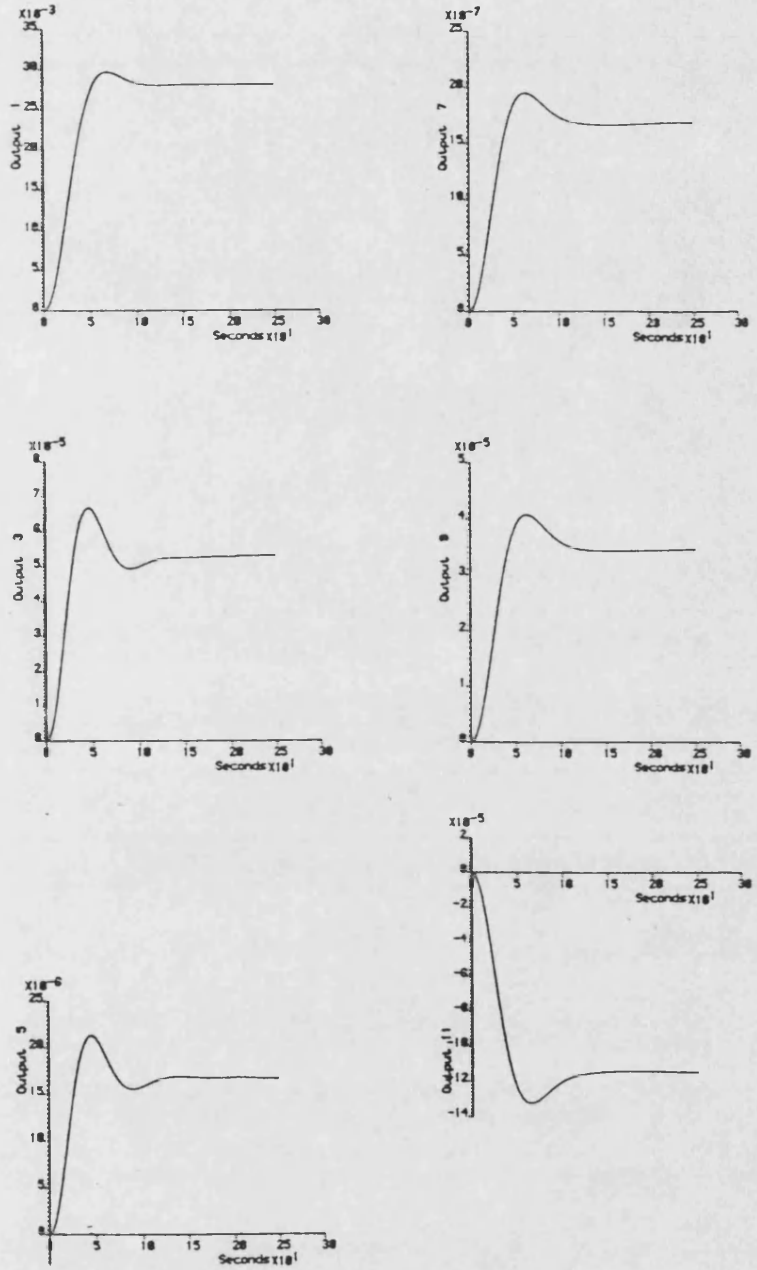


Figure 9.10: Step responses of input 1 of nominal Platform with 26th order controller

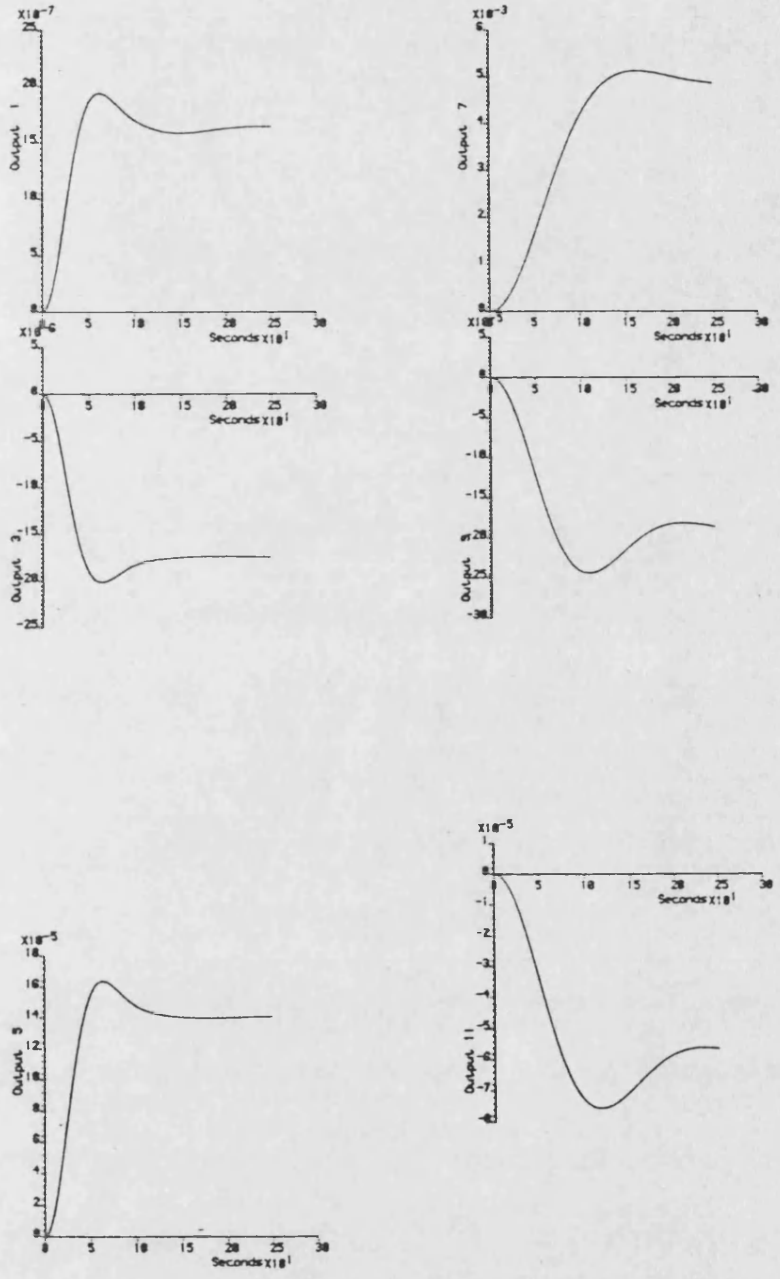


Figure 9.11: Step responses of input 4 of nominal Platform with 26th order controller

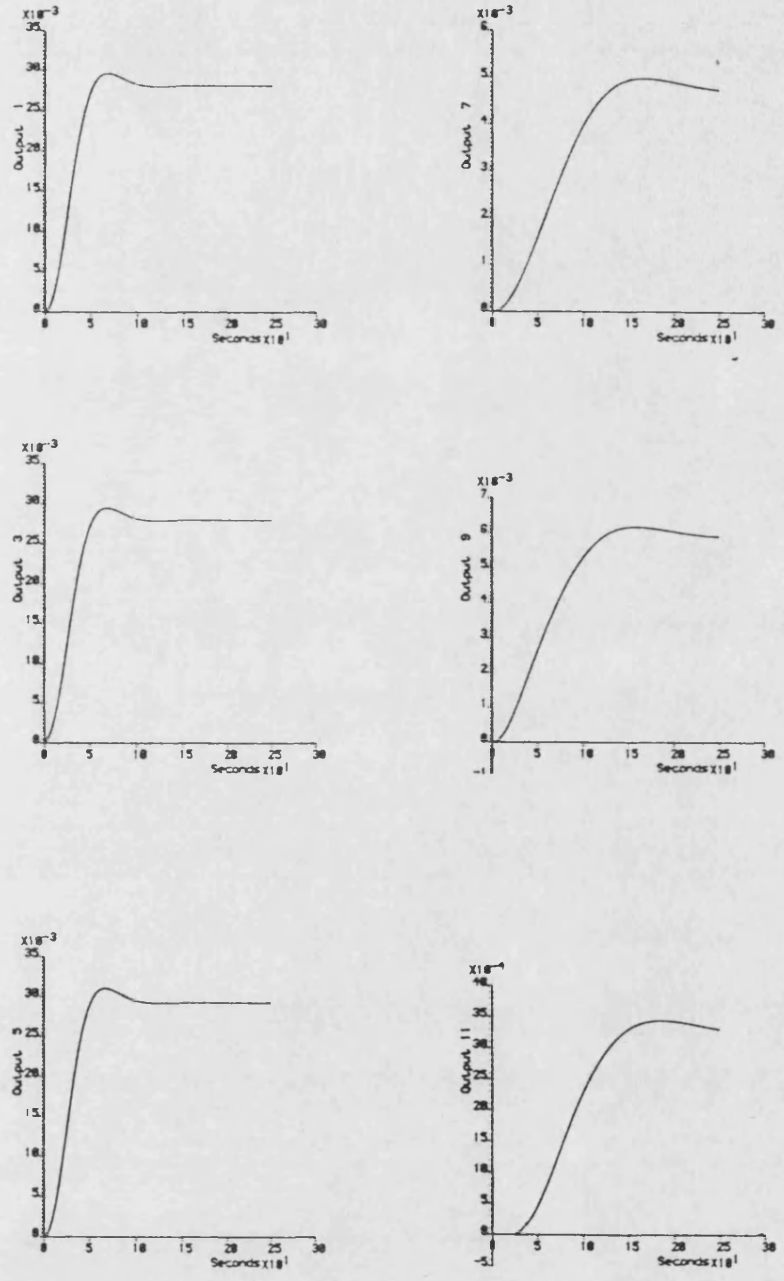


Figure 9.12: Step responses of all inputs of nominal Platform with 26th order controller

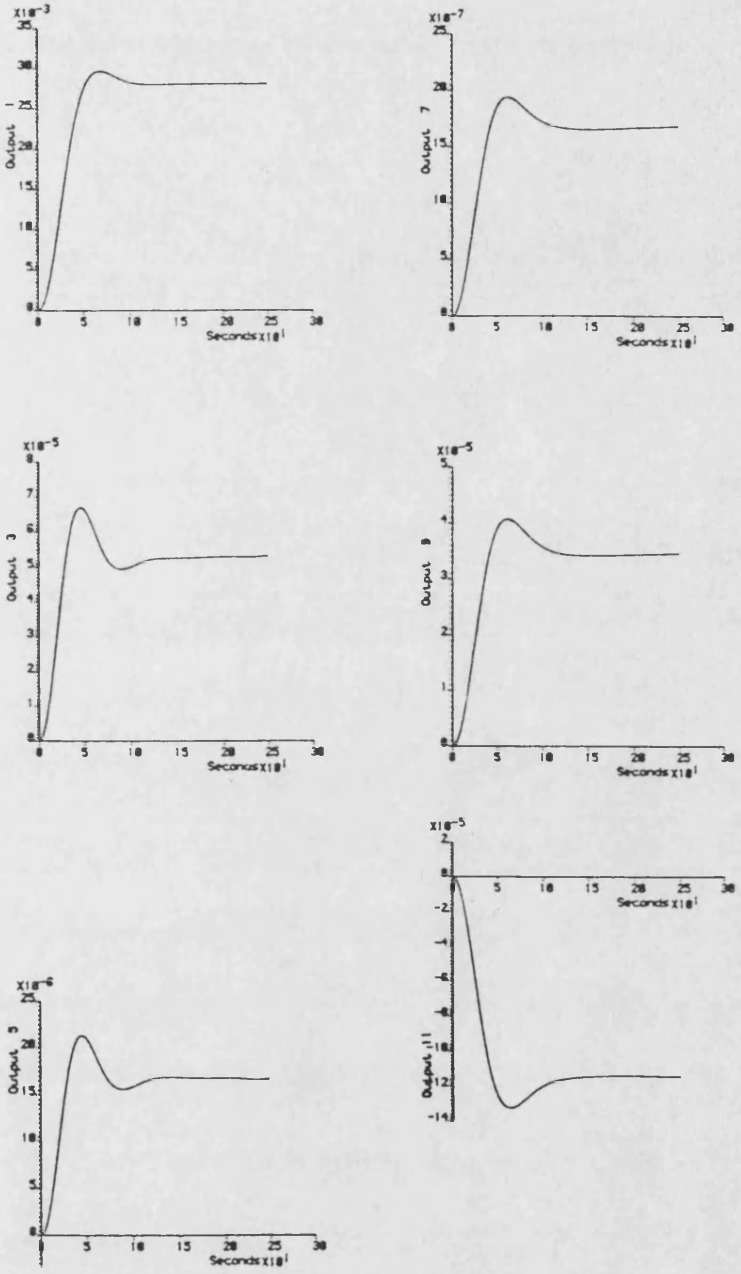


Figure 9.13: Step responses of input 1 of perturbed Platform with 26th order controller

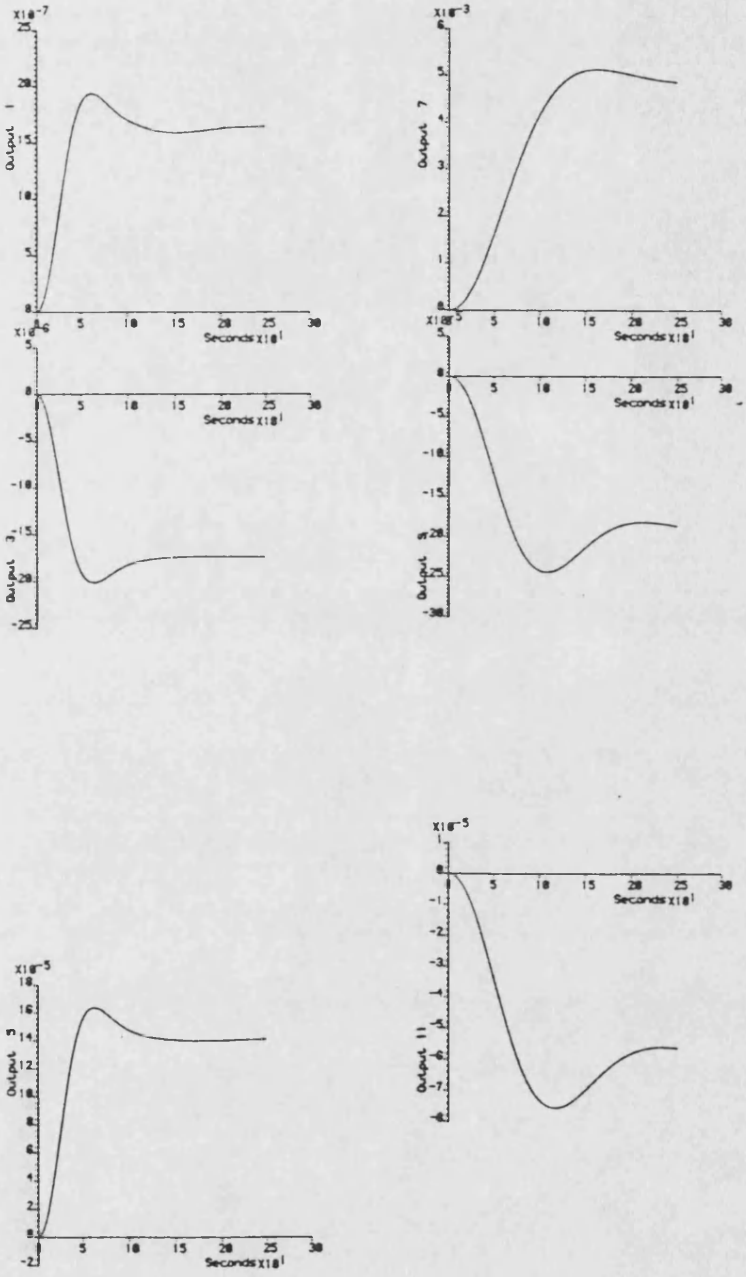


Figure 9.14: Step responses of input 4 of perturbed Platform with 26th order controller

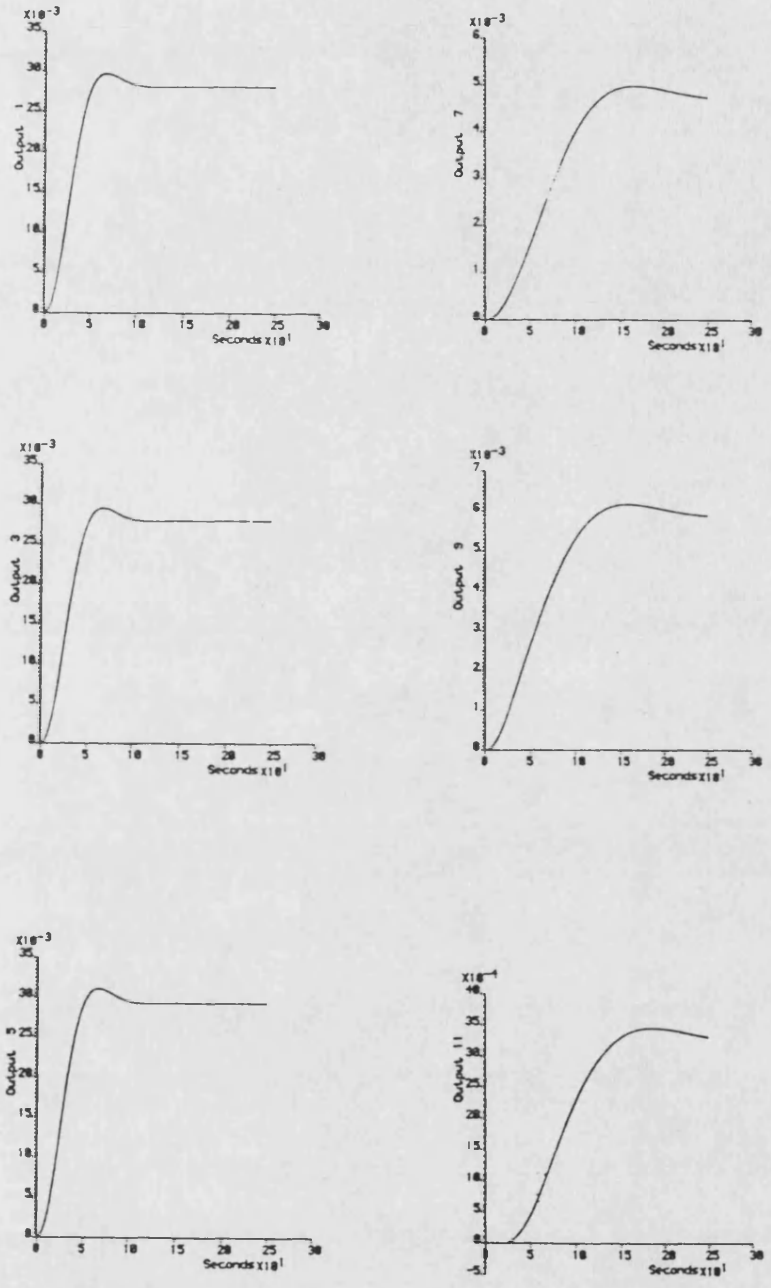


Figure 9.15: Step responses of all inputs of perturbed Platform with 26th order controller

loop poles of the complete closed loop system are shown in table 9.10. The results of the simulation of this system are shown in figures 9.16, 9.17, and 9.18, where again it can be seen that no noticeable degradation of the system performance over the full 52nd order controller occurs. Similarly, no noticeable degradation in the performance can be seen in the simulation results of the perturbed model with this 18th order controller as shown in figures 9.19, 9.20, and 9.21.

A further reduction was attempted by forming a model from the rigid modes alone, the resulting model being 12th order. A 12th order controller was designed using the program LQGDES again with unit weighting matrices, and the poles of the complete closed loop system are shown in table 9.11. It can be seen that this design suffers catastrophically from "spillover" and is unstable.

These cases highlight a difficulty with the use of this technique in that it is not possible to define what level of maximum model error is acceptable for the resulting controller design to produce a stable closed loop system. Besides being dependent on the actual model itself, it is also dependent on the required performance of the closed loop system, as will be shown later.

9.3.3 Controller Design—Controller Order Reduction

Again in the same manner as was used for the experimental beam controller, a full order controller was designed and then it was attempted to derive an "equivalent" reduced order controller using the program QCOVER. The full order controller used as the basis for this approach was the 52nd order baseline controller introduced in Section 9.3.1.

The program QCOVER was run with a variety of output covariance matrices but in all cases it terminated due to numerical difficulties, after deriving the 15th order "equivalent" controller, and none of the reduced order controllers resulted in a stable closed loop system.

9.3.4 Output Feedback Controller

As an alternative to the above controllers, which are all based on state feedback, a controller based on output feedback was designed in a similar manner to that used in the previous chapter for the experimental beam. The method of independent modal space control is even less suited to the high order problem considered here than to the experimental beam and so

Designed Eigenvalues Of Plant	
Real	Imaginary
-0.18623	37.235
-0.18623	-37.235
-0.86722E-01	17.341
-0.86722E-01	-17.341
-0.50830E-01	10.159
-0.50830E-01	-10.159
-0.21624E-01	0.21613E-01
-0.21624E-01	-0.21613E-01
-0.24439E-01	0.24425E-01
-0.24439E-01	-0.24425E-01
-0.25966E-01	0.25949E-01
-0.25966E-01	-0.25949E-01
-0.59839E-01	0.59626E-01
-0.59839E-01	-0.59626E-01
-0.59645E-01	0.59434E-01
-0.59645E-01	-0.59434E-01
-0.59442E-01	0.59233E-01
-0.59442E-01	-0.59233E-01

Designed Eigenvalues Of Estimator	
Real	Imaginary
-0.21857	37.235
-0.21857	-37.235
-0.93103E-01	17.341
-0.93103E-01	-17.341
-0.53806E-01	10.159
-0.53806E-01	-10.159
-0.59946E-01	0.59520E-01
-0.59946E-01	-0.59520E-01
-0.59751E-01	0.59328E-01
-0.59751E-01	-0.59328E-01
-0.59547E-01	0.59129E-01
-0.59547E-01	-0.59129E-01
-0.25975E-01	0.25940E-01
-0.25975E-01	-0.25940E-01
-0.24447E-01	0.24418E-01
-0.24447E-01	-0.24418E-01
-0.21629E-01	0.21608E-01
-0.21629E-01	-0.21608E-01

Eigenvalues Of Closed Loop System	
Real	Imaginary
-0.31205	62.412
-0.31205	-62.412
-0.28005	56.006
-0.28005	-56.006
-0.19620	39.245
-0.19620	-39.245
-0.18623	37.235
-0.18623	-37.235
-0.21857	37.235
-0.21857	-37.235
-0.15470	30.939
-0.15470	-30.939
-0.14870	29.742
-0.14870	-29.742
-0.14745	29.486
-0.14745	-29.486
-0.13250	26.502
-0.13250	-26.502
-0.12410	24.818
-0.12410	-24.818
-0.10390	20.780
-0.10390	-20.780
-0.86722E-01	17.341
-0.86722E-01	-17.341
-0.93103E-01	17.341
-0.93103E-01	-17.341
-0.39149E-01	7.8328
-0.39149E-01	-7.8328
-0.40149E-01	8.0316
-0.40149E-01	-8.0316
-0.43950E-01	8.7887
-0.43950E-01	-8.7887
-0.48198E-01	9.6394
-0.48198E-01	-9.6394
-0.47300E-01	9.4591
-0.47300E-01	-9.4591
-0.46250E-01	9.2507
-0.46250E-01	-9.2507
-0.50830E-01	10.159
-0.50830E-01	-10.159
-0.53806E-01	10.159
-0.53806E-01	-10.159
-0.16799E-01	3.3597
-0.16799E-01	-3.3597
-0.15296E-01	3.0625
-0.15296E-01	-3.0625
-0.21455E-01	0.21651E-01
-0.21455E-01	-0.21651E-01
-0.21798E-01	0.21569E-01
-0.21798E-01	-0.21569E-01
-0.25877E-01	0.25966E-01
-0.25877E-01	-0.25966E-01
-0.26064E-01	0.25921E-01
-0.26064E-01	-0.25921E-01
-0.24373E-01	0.24438E-01
-0.24373E-01	-0.24438E-01
-0.24514E-01	0.24404E-01
-0.24514E-01	-0.24404E-01
-0.58974E-01	0.59327E-01
-0.58974E-01	-0.59327E-01
-0.59595E-01	0.59653E-01
-0.59595E-01	-0.59653E-01
-0.59225E-01	0.59482E-01
-0.59225E-01	-0.59482E-01
-0.60192E-01	0.59491E-01
-0.60192E-01	-0.59491E-01
-0.60197E-01	0.59228E-01
-0.60197E-01	-0.59228E-01
-0.59995E-01	0.59075E-01
-0.59995E-01	-0.59075E-01

Table 9.10: Closed loop poles of platform with 18th order controller

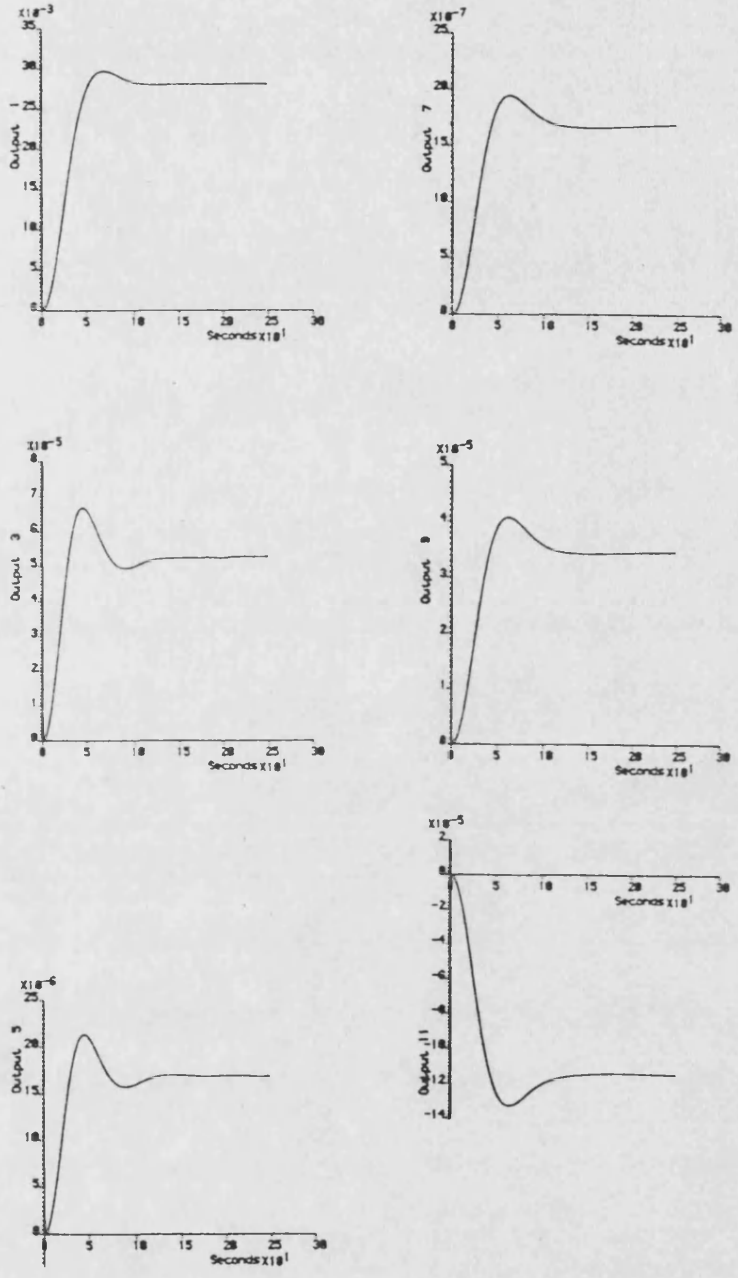


Figure 9.16: Step responses of input 1 of nominal Platform with 18th order controller

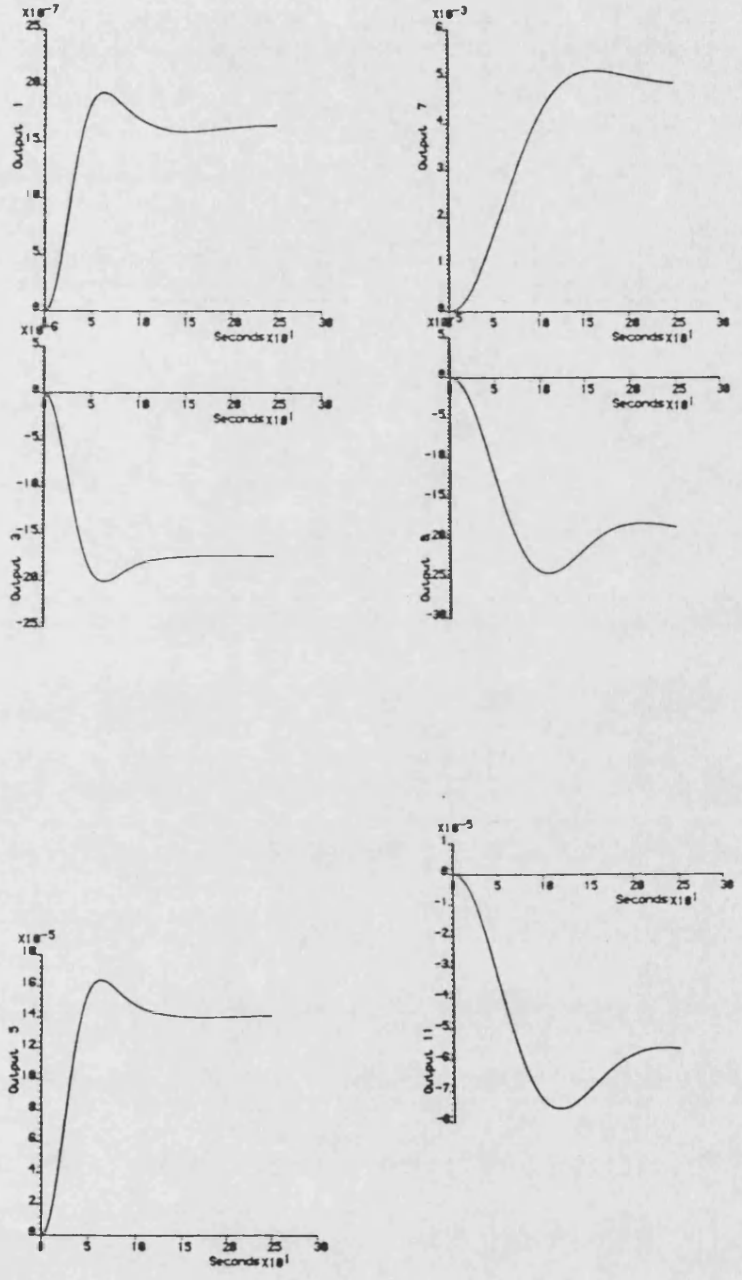


Figure 9.17: Step responses of input 4 of nominal Platform with 18th order controller

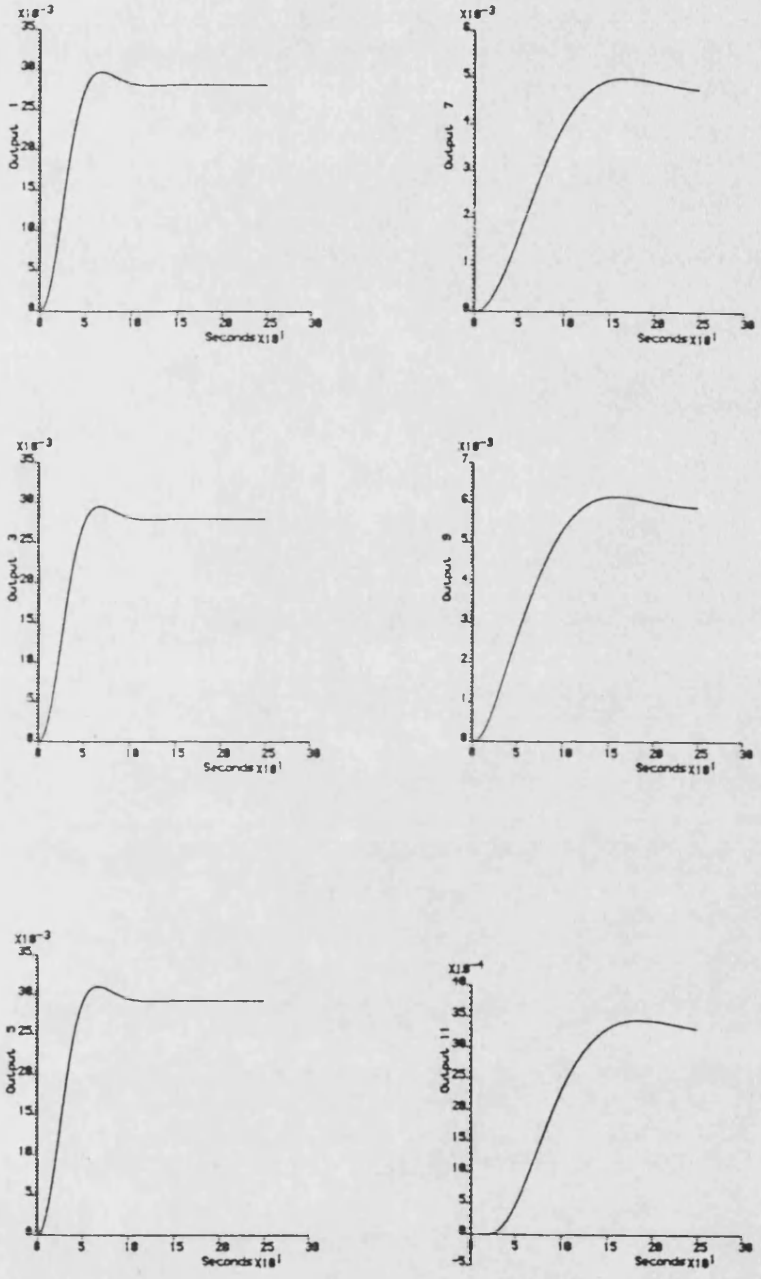


Figure 9.18: Step responses of all inputs of nominal Platform with 18th order controller

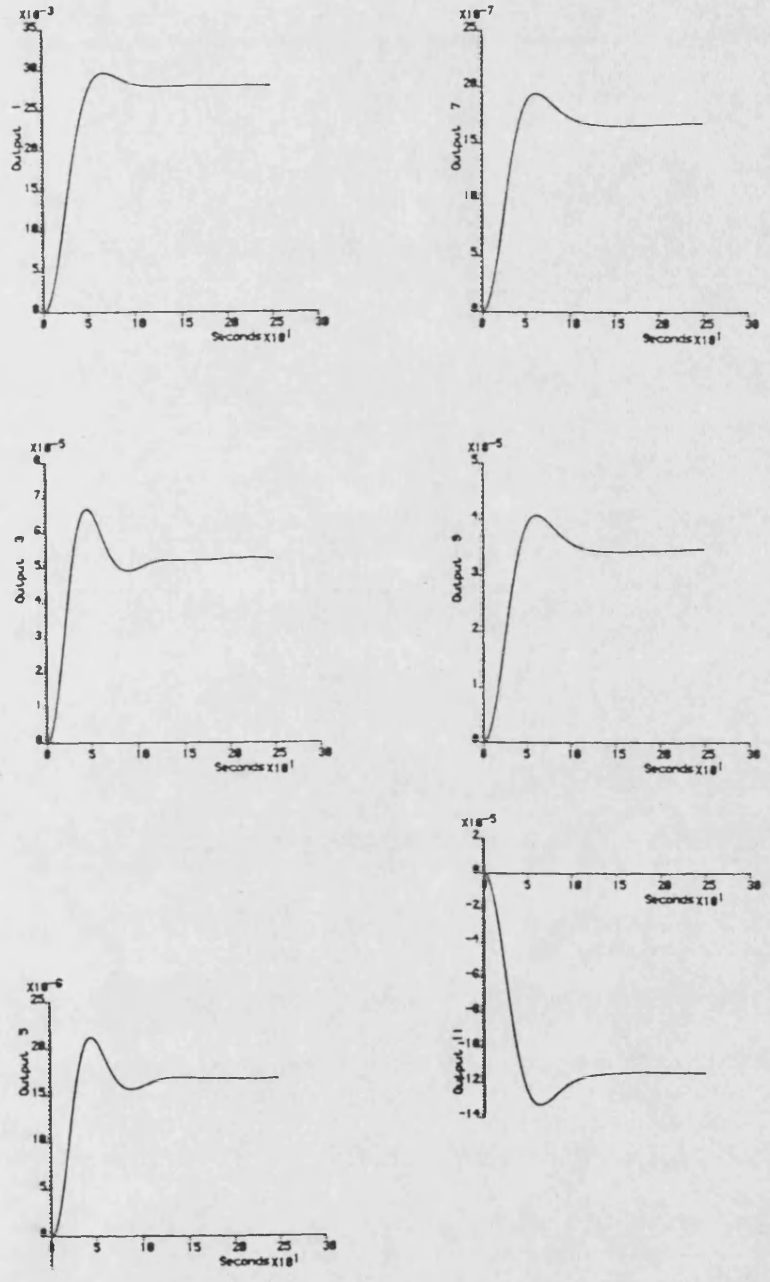


Figure 9.19: Step responses of input 1 of perturbed Platform with 18th order controller

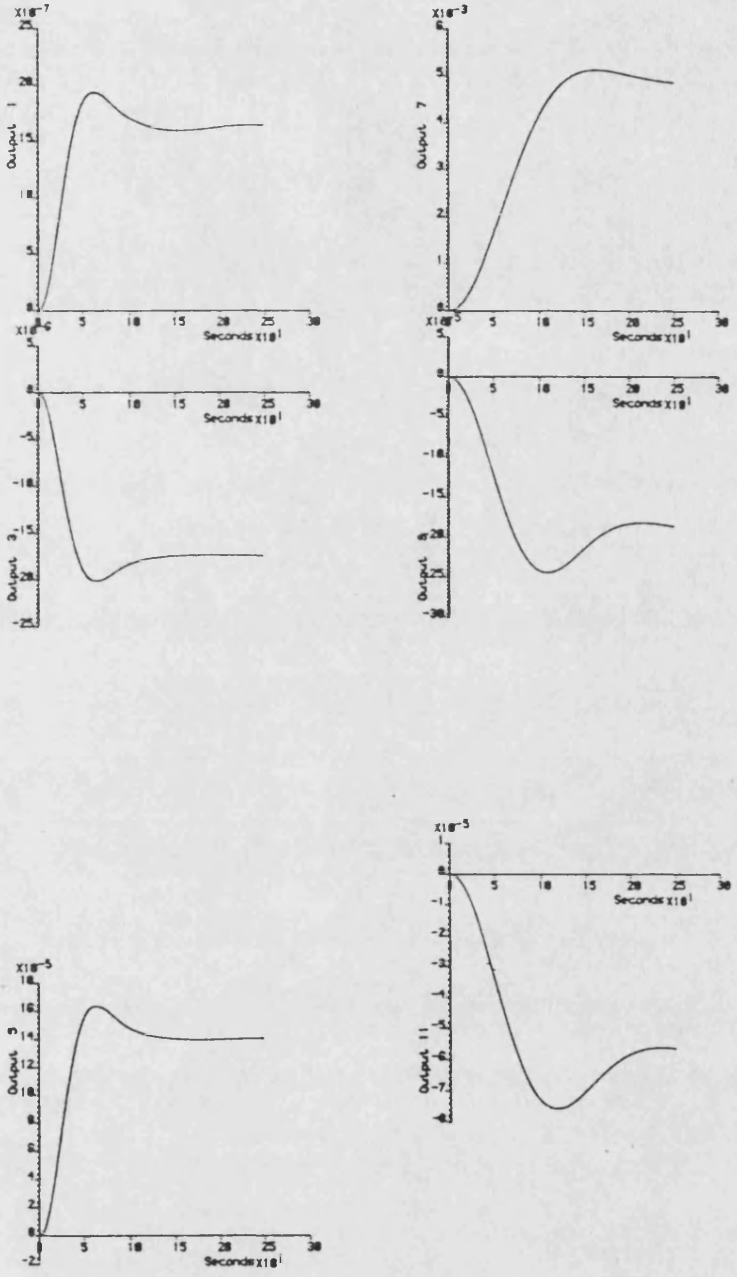


Figure 9.20: Step responses of input 4 of perturbed Platform with 18th order controller

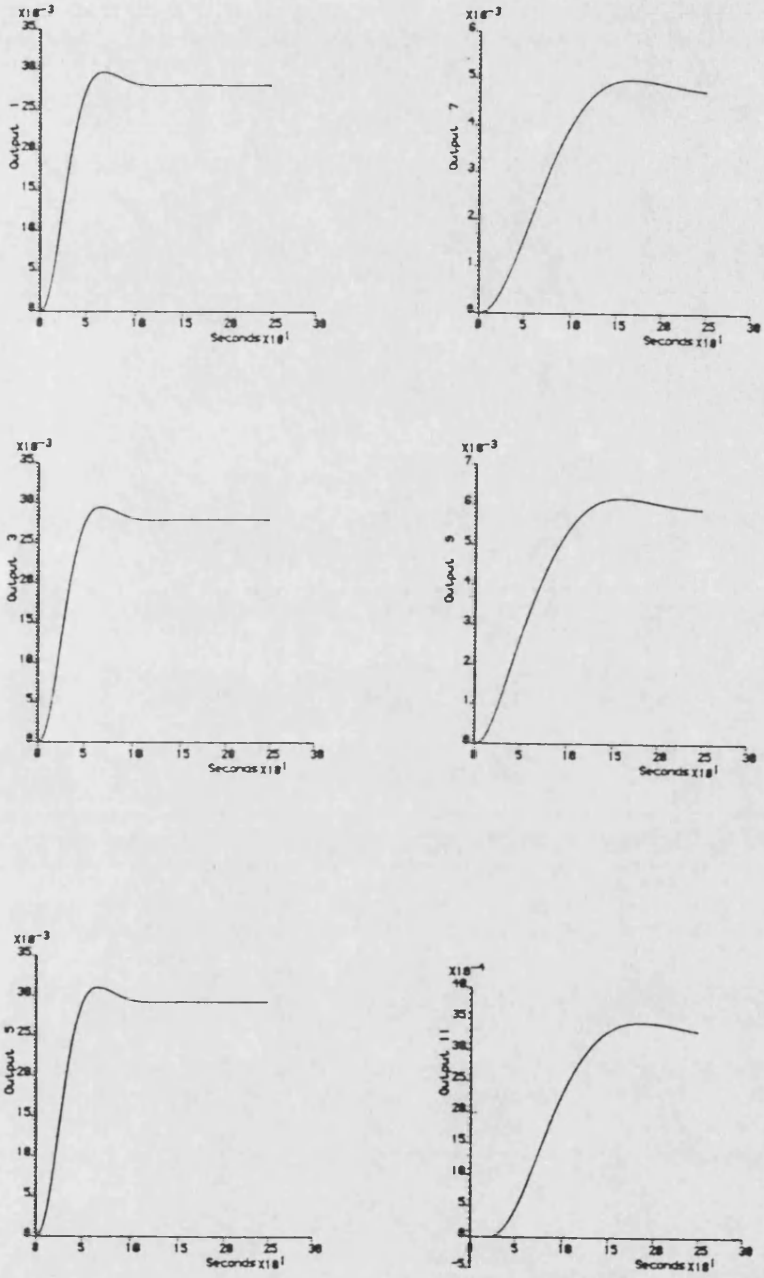


Figure 9.21: Step responses of all inputs of perturbed Platform with 18th order controller

Designed Eigenvalues Of Plant	
Real	Imaginary
-0.21624E-01	0.21613E-01
-0.21624E-01	-0.21613E-01
-0.25966E-01	0.25949E-01
-0.25966E-01	-0.25949E-01
-0.24439E-01	0.24425E-01
-0.24439E-01	-0.24425E-01
-0.59839E-01	0.59626E-01
-0.59839E-01	-0.59626E-01
-0.59645E-01	0.59434E-01
-0.59645E-01	-0.59434E-01
-0.59442E-01	0.59233E-01
-0.59442E-01	-0.59233E-01

Designed Eigenvalues Of Estimator	
Real	Imaginary
-0.59946E-01	0.59520E-01
-0.59946E-01	-0.59520E-01
-0.59751E-01	0.59328E-01
-0.59751E-01	-0.59328E-01
-0.59547E-01	0.59129E-01
-0.59547E-01	-0.59129E-01
-0.25975E-01	0.25940E-01
-0.25975E-01	-0.25940E-01
-0.24447E-01	0.24418E-01
-0.24447E-01	-0.24418E-01
-0.21629E-01	0.21608E-01
-0.21629E-01	-0.21608E-01

Eigenvalues Of Closed Loop System	
Real	Imaginary
-0.31205	62.412
-0.31205	-62.412
-0.28005	56.006
-0.28005	-56.006
-0.19620	39.245
-0.19620	-39.245
-0.18620	37.235
-0.18620	-37.235
-0.15470	30.939
-0.15470	-30.939
-0.14870	29.742
-0.14870	-29.742
-0.14745	29.486
-0.14745	-29.486
-0.13250	26.502
-0.13250	-26.502
-0.12410	24.818
-0.12410	-24.818
-0.10390	20.780
-0.10390	-20.780
-0.86700E-01	17.341
-0.86700E-01	-17.341
-0.50800E-01	10.159
-0.50800E-01	-10.159
-0.48200E-01	9.6394
-0.48200E-01	-9.6394
-0.47300E-01	9.4591
-0.47300E-01	-9.4591
-0.46250E-01	9.2507
-0.46250E-01	-9.2507
-0.43950E-01	8.7887
-0.43950E-01	-8.7887
-0.40150E-01	8.0316
-0.40150E-01	-8.0316
-0.39150E-01	7.8328
-0.39150E-01	-7.8328
-0.15224E-01	3.0625
-0.15224E-01	-3.0625
-0.16800E-01	3.3597
-0.16800E-01	-3.3597
-0.93419E-01	0.00000E+00
0.93465E-01	0.00000E+00
0.82951E-01	0.00000E+00
-0.82942E-01	0.00000E+00
-0.38859E-01	0.00000E+00
0.35460E-01	0.00000E+00
0.38731E-01	0.00000E+00
-0.35492E-01	0.00000E+00
-0.89271E-09	0.00000E+00
0.89256E-09	0.00000E+00
-0.75555E-18	0.14749E-09
-0.75555E-18	-0.14749E-09
-0.22333E-16	0.00000E+00
0.18581E-16	0.45640E-11
0.18581E-16	-0.45640E-11
-0.26021E-16	0.00000E+00
0.78190E-21	0.00000E+00
0.78190E-21	0.00000E+00
0.00000E+00	0.00000E+00
0.00000E+00	0.00000E+00
0.00000E+00	0.00000E+00
0.00000E+00	0.00000E+00
0.00000E+00	0.00000E+00
0.00000E+00	0.00000E+00

Table 9.11: Closed loop poles of platform with 12th order controller

is not pursued. Thus the output feedback controller was designed using the program RMDES (where the elastic modes are ignored), such that the poles corresponding to the rigid modes are located in the same positions as the poles of the rigid modes defined by the baseline 52nd order controller (see Section 9.3.1). The poles of the resulting closed loop system are shown in table 9.12.

The simulated step responses of the resulting system for the three input cases are shown in figures 9.22, 9.23, and 9.24, from which it can be seen that the responses are slightly slower than with the state feedback based controllers but are more damped. The simulated step responses of the perturbed model with this controller are shown in figures 9.25, 9.26, and 9.27, which show no noticeable degradation over the nominal case.

An interesting facet of these step responses is the interaction dynamics. With the output feedback based controller the interaction tends to steady state values of zero, but with the state feedback based controllers described earlier, the interaction terms have finite steady state values, and also generally have larger transients.

9.3.5 Remarks

The previous sections have shown that it is possible to design controllers, both state feedback based and output feedback based, for realistic models of large flexible spacecraft. Although the design of a six degree of freedom controller has been informative, for instance to examine the interaction between the translational and rotational degrees of freedom, the above exercise has several limitations.

Firstly, the consideration of the problem of positioning and pointing of a spacecraft in 3-dimensional space cannot be treated in this simple manner. Although the attitude of a 3-axis stabilised spacecraft (the direction in which it "points") can, and generally is, defined in terms of a set of three rotational angles (the rotational degrees of freedom), the position of the spacecraft in space is a function of the orbit dynamics. (For an introduction to orbit dynamics, see [7] or [114].) Generally the orbit is defined by the operational requirements of the spacecraft and thus are not defined arbitrarily. Although the attitude dynamics and the orbit dynamics are coupled, they are often considered to be decoupled as the coupling is generally small, and the attitude dynamics are usually much faster than the orbit dynamics.

Secondly, the response time of the controllers designed above has deliberately been made fairly slow (of the order of 50–100 seconds), so that the

Eigenvalues Of Closed Loop System		
Real	Imaginary	
-0.32895	62.412	
-0.32895	-62.412	
-0.33015	56.005	
-0.33015	-56.005	
-0.20809	39.245	
-0.20809	-39.245	
-0.21362	37.235	
-0.21362	-37.235	
-0.19475	30.938	
-0.19475	-30.938	
-0.15169	29.742	
-0.15169	-29.742	
-0.14759	29.486	
-0.14759	-29.486	
-0.13911	26.502	
-0.13911	-26.502	
-0.13370	24.818	
-0.13370	-24.818	
-0.11586	20.780	
-0.11586	-20.780	
-0.16391	17.340	
-0.16391	-17.340	
-0.58806E-01	10.159	
-0.58806E-01	-10.159	
-0.72217E-01	9.6389	
-0.72217E-01	-9.6389	
-0.59527E-01	9.4587	
-0.59527E-01	-9.4587	
-0.48495E-01	9.2508	
-0.48495E-01	-9.2508	
-0.46990E-01	8.7887	
-0.46990E-01	-8.7887	
-0.61366E-01	7.8329	
-0.61366E-01	-7.8329	
-0.54638E-01	8.0315	
-0.54638E-01	-8.0315	
-0.30965E-01	3.3594	
-0.30965E-01	-3.3594	
-0.16447E-01	3.0625	
-0.16447E-01	-3.0625	
-0.55805E-01	0.59298E-01	Designed Eigenvalues Of Plant
-0.55805E-01	-0.59298E-01	
-0.59851E-01	0.59475E-01	
-0.59851E-01	-0.59475E-01	
-0.64915E-01	0.59491E-01	
-0.64915E-01	-0.59491E-01	
-0.24424E-01	0.25155E-01	
-0.24424E-01	-0.25155E-01	
-0.24462E-01	0.25076E-01	
-0.24462E-01	-0.25076E-01	
-0.24500E-01	0.24995E-01	
-0.24500E-01	-0.24995E-01	

Table 9.12: Closed loop poles of platform with output feedback controller

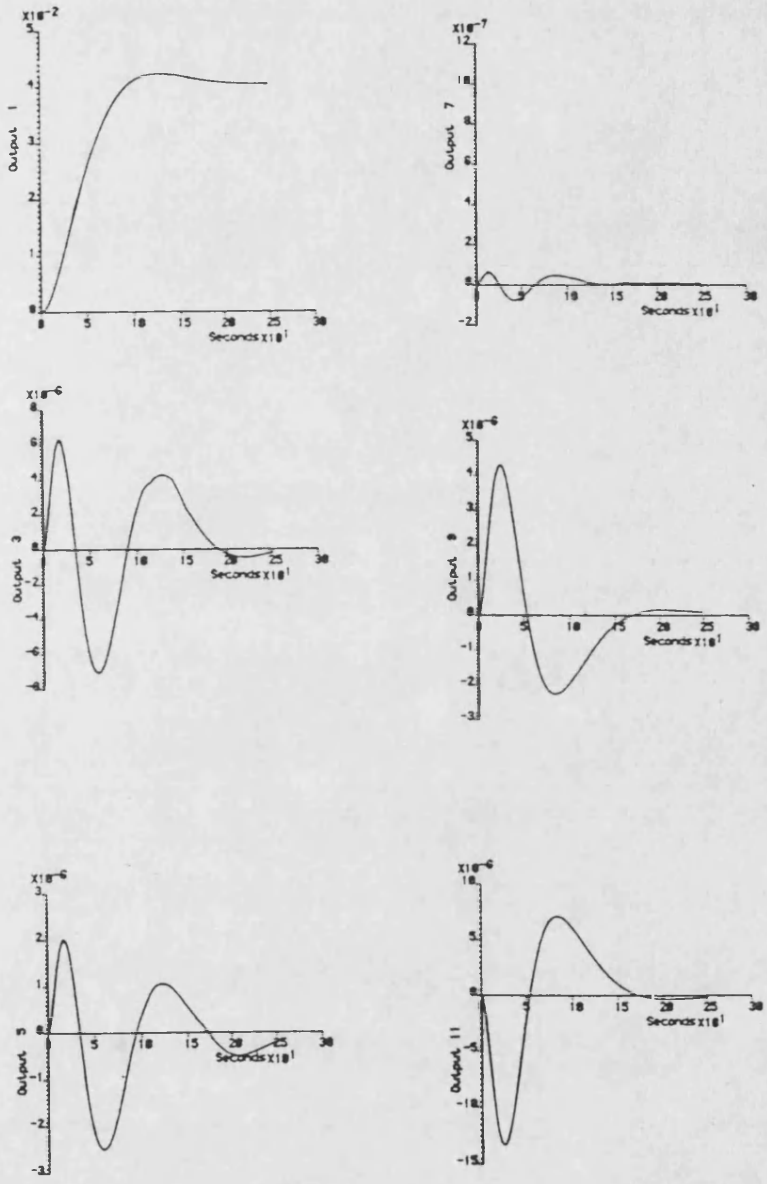


Figure 9.22: Step responses of input 1 of nominal Platform with output feedback controller

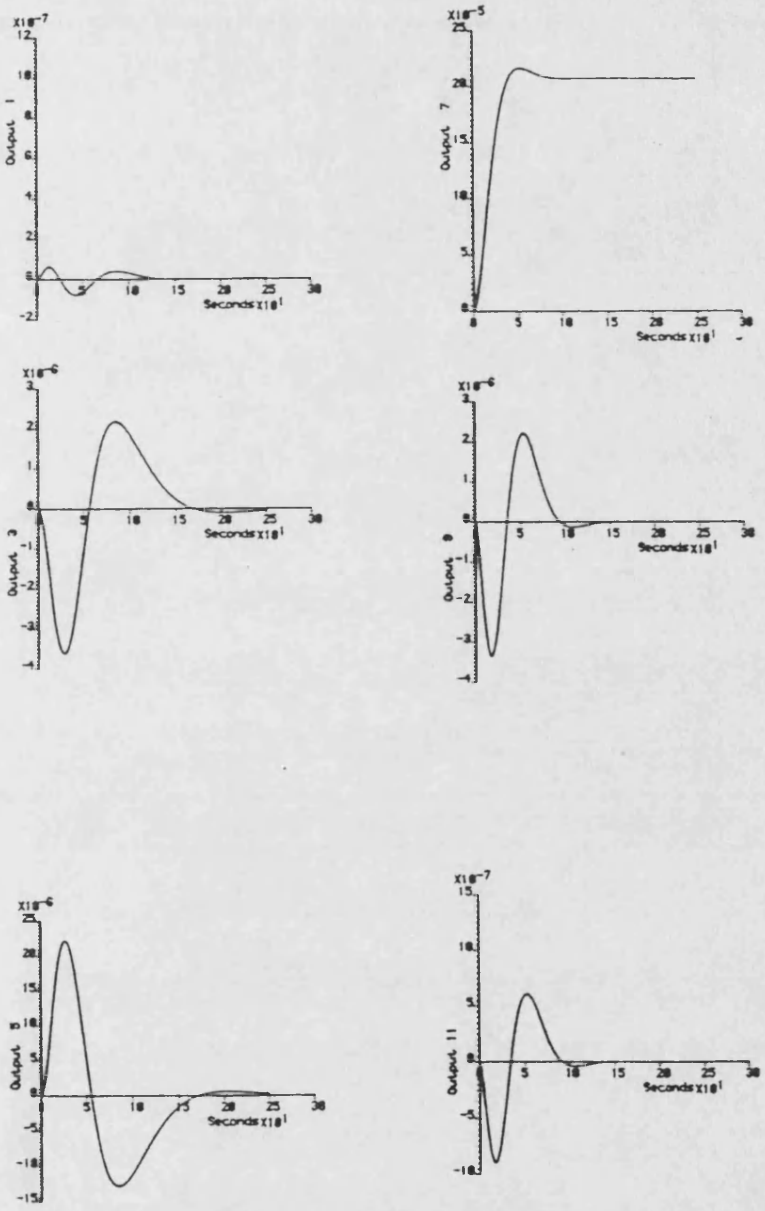


Figure 9.23: Step responses of input 4 of nominal Platform with output feedback controller

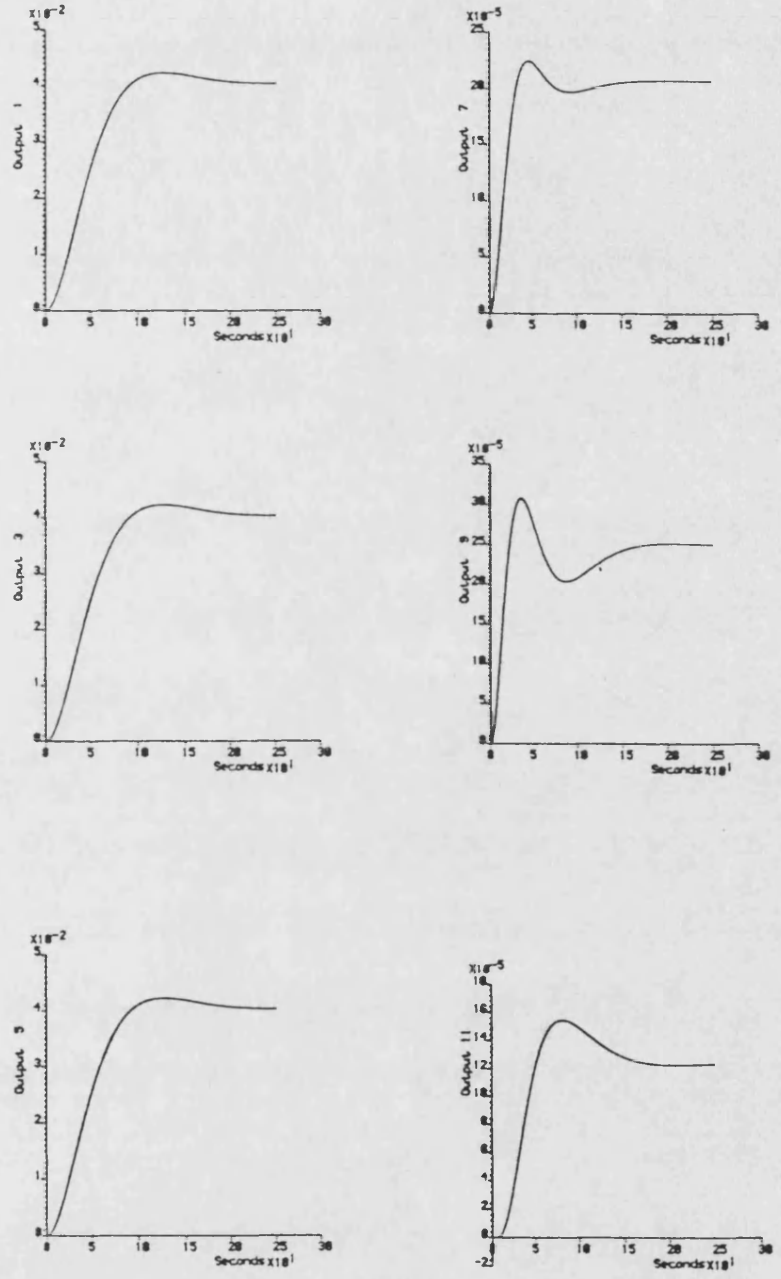


Figure 9.24: Step responses of all inputs of nominal Platform with output feedback controller

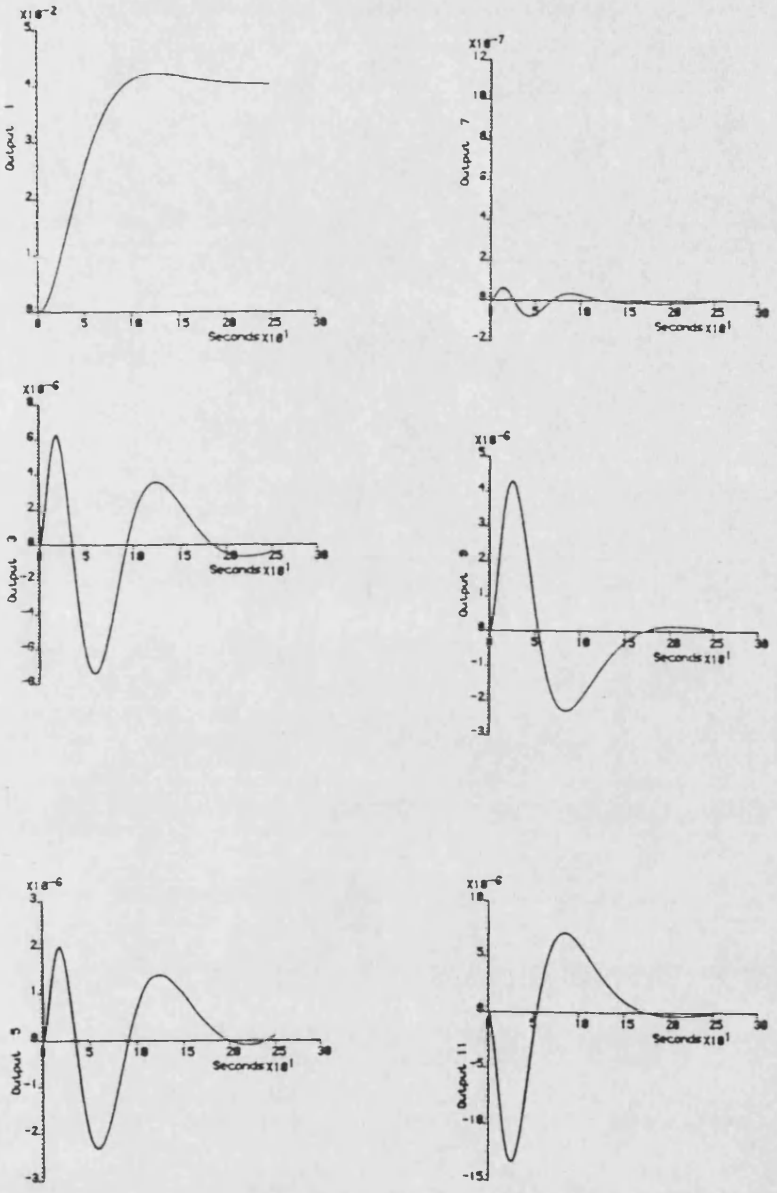


Figure 9.25: Step responses of input 1 of perturbed Platform with output feedback controller

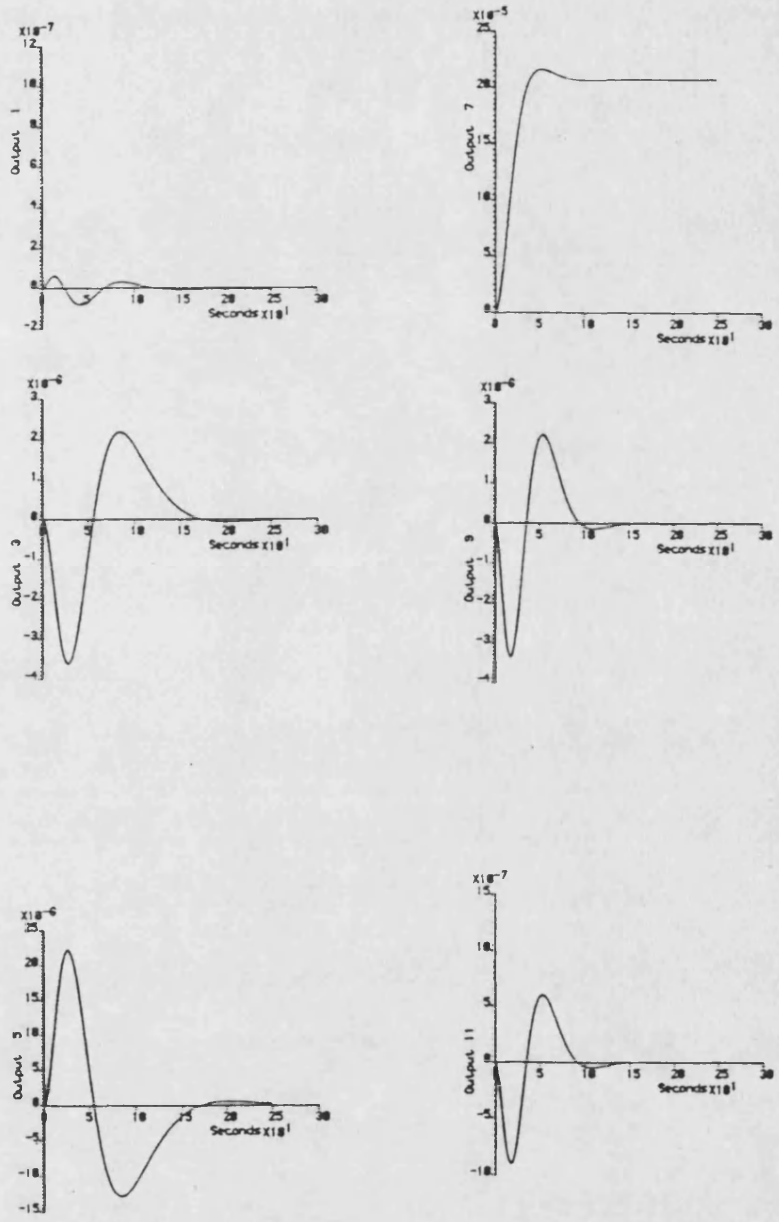


Figure 9.26: Step responses of input 4 of perturbed Platform with output feedback controller

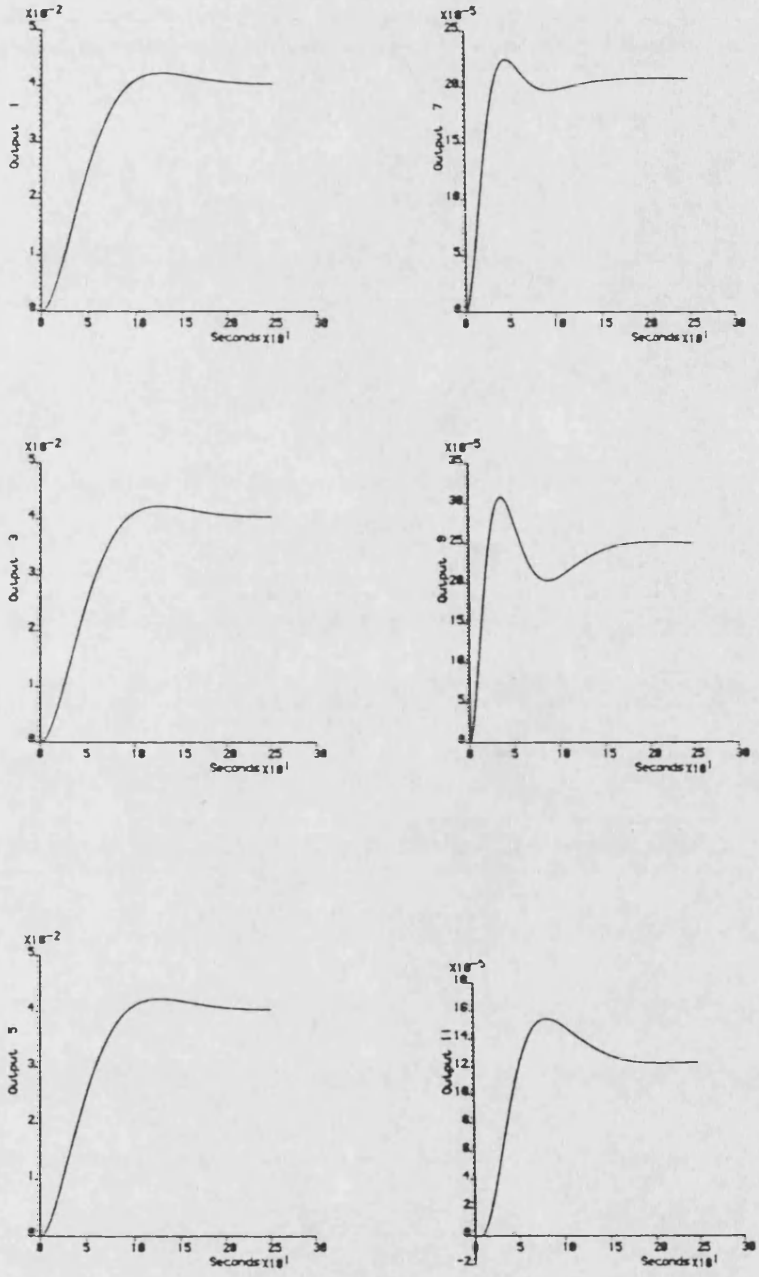


Figure 9.27: Step responses of all inputs of perturbed Platform with output feedback controller

controller bandwidth was fairly narrow to demonstrate how it is possible to reduce the interaction with the elastic modes at the expense of response times. Obviously this is essentially avoiding the interaction problem rather than dealing with it, and it could be expected that a "faster" controller might be more difficult to design.

Thirdly, the perturbed model used for "robustness" evaluation appears to be of minimal value as no noticeable performance degradation was observed, even in a controller order change from 52nd to 18th order.

For the above reasons, a second controller design exercise was carried out based on the platform model, but this time the more realistic case of a three degree of freedom attitude controller was considered, of the type required for 3-axis stabilisation of the spacecraft, and a much faster response time was sought such that the frequencies of some of the elastic modes would lie within the controller bandwidth. Also the use of the perturbed model for attempting to compare the "robustness" of various controller designs was dropped.

9.4 Attitude Controller

In this section the problem of designing an attitude controller for the platform spacecraft is considered. That is, the problem of designing a controller which will produce three-axis stabilisation of the spacecraft. The state space model is formulated from the same modal data as presented in Section 9.2, but the three translational rigid modes have been ignored, and thus the resulting model is 46th order. This model is referred to as the platform attitude model.

Note that this model also incorporates additional terms in the output matrix corresponding to the three components of rotation of node 26 of the finite element representation (see figure 9.2). Node 26 corresponds to the interface between one of the payloads (in fact the payload most distant from the utilities module) and the spacecraft. The reason for this addition is to highlight one of the advantages of the finite element representation. Frequently, the users of payload capability on spacecraft require to know the pointing accuracy not just of the main spacecraft, but also of their payload. Thus the motion of the payload interface corresponding to node 26, relative to the main spacecraft motion (that is node 1), can be examined for satisfactory behaviour. This capability is obviously most useful in a case such as this where the structure between the payload interface and the point

of attitude control (node 1) exhibits significant flexibility. This allows the assessment of the disturbances that payloads, or other parts of the structure of interest, undergo during spacecraft operations.

In a similar manner to that described for the previous controller design exercise, a "baseline" controller is designed to act as a reference point for subsequent controller designs using full 46th order state feedback and a 46th order state estimator. The design of this baseline controller is now discussed.

9.4.1 Baseline Controller

The baseline controller was designed using the program LQGDES with diagonal weighting matrices. The weighting terms corresponding to the elastic modes were set to unity for both the state feedback matrix computation and the state estimator gain matrix computation, but the weighting terms corresponding to the rigid modes were set very high, thus causing a good deal of movement from their open loop locations to their closed loop locations. This was done deliberately to position the poles of the rigid modes amongst the poles of the elastic modes. This can be seen in table 9.13 where the closed loop plant poles and the state estimator poles are shown.

The closed loop system comprised of the platform attitude model and this controller was simulated, and subjected to three test cases, these being a unit step demand on each input independently. These three tests cases are used throughout this exercise.

The three attitude angle responses for a unit step demand on the first input are shown in figure 9.28, the rotations of node 26 are shown in figure 9.29, and the error signals are shown in figure 9.30. The same three sets of responses for a unit step demand on the second input are shown in figures 9.31, 9.32, and 9.33, and the responses for a unit step demand on the third input are shown in figures 9.34, 9.35, and 9.36.

An obvious difference between the set of responses for this controller and those for the previous controller design exercise is the much greater contribution of the elastic modes to the outputs for the controller presented here. Also, the rigid mode response times are an order of magnitude faster (this is best seen from the error responses). Notice also that the level of interaction between the inputs is much greater.

The acceptance of the oscillations present in the output responses for a realistic example could only be decided by the engineering requirements for that example. No attempt has been made with this controller to increase the level of damping of any of the elastic modes, although this could be

Designed Eigenvalues Of Plant		Designed Eigenvalues Of Estimator	
Real	Imaginary	Real	Imaginary
-0.31205	62.412	-330.57	0.49971
-0.31205	-62.412	-330.57	-0.49971
-0.28005	56.006	-0.31324	62.412
-0.28005	-56.006	-0.31324	-62.412
-0.19620	39.245	-0.28047	56.006
-0.19620	-39.245	-0.28047	-56.006
-0.18620	37.235	-26.159	0.49986
-0.18620	-37.235	-26.159	-0.49986
-0.15470	30.939	-0.19699	39.245
-0.15470	-30.939	-0.19699	-39.245
-0.14870	29.742	-0.18641	37.235
-0.14870	-29.742	-0.18641	-37.235
-0.14745	29.486	-0.16205	30.939
-0.14745	-29.486	-0.16205	-30.939
-0.13250	26.502	-0.14892	29.742
-0.13250	-26.502	-0.14892	-29.742
-0.12410	24.818	-0.14745	29.486
-0.12410	-24.818	-0.14745	-29.486
-0.10390	20.780	-0.13263	26.502
-0.10390	-20.780	-0.13263	-26.502
-0.86703E-01	17.341	-0.12411	24.818
-0.86703E-01	-17.341	-0.12411	-24.818
-4.0655	4.0656	-0.10422	20.780
-4.0655	-4.0656	-0.10422	-20.780
-0.50800E-01	10.159	-0.86744E-01	17.341
-0.50800E-01	-10.159	-0.86744E-01	-17.341
-0.48201E-01	9.6394	-0.62501	0.39396
-0.48201E-01	-9.6394	-0.62501	-0.39396
-0.47301E-01	9.4591	-0.16804E-01	3.3597
-0.47301E-01	-9.4591	-0.16804E-01	-3.3597
-0.46250E-01	9.2507	-0.15300E-01	3.0625
-0.46250E-01	-9.2507	-0.15300E-01	-3.0625
-0.43950E-01	8.7887	-0.39223E-01	7.8328
-0.43950E-01	-8.7887	-0.39223E-01	-7.8328
-0.40151E-01	8.0316	-0.40229E-01	8.0316
-0.40151E-01	-8.0316	-0.40229E-01	-8.0316
-0.39151E-01	7.8328	-0.50815E-01	10.159
-0.39151E-01	-7.8328	-0.50815E-01	-10.159
-1.1435	1.1436	-0.43991E-01	8.7887
-1.1435	-1.1436	-0.43991E-01	-8.7887
-0.16802E-01	3.3597	-0.48309E-01	9.6394
-0.16802E-01	-3.3597	-0.48309E-01	-9.6394
-0.15300E-01	3.0625	-0.47309E-01	9.4591
-0.15300E-01	-3.0625	-0.47309E-01	-9.4591
-0.15622	0.15631	-0.46252E-01	9.2507
-0.15622	-0.15631	-0.46252E-01	-9.2507

Table 9.13: Closed loop poles of platform with initial baseline controller

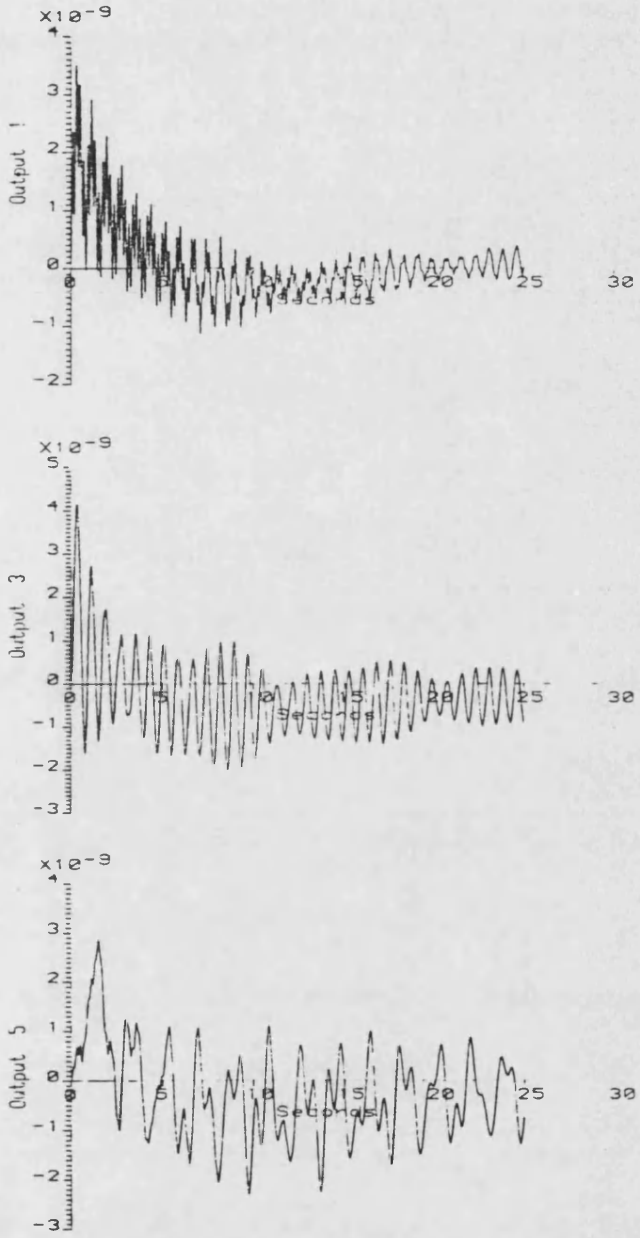


Figure 9.28: Attitude responses of input 1 of Platform attitude model with initial baseline controller

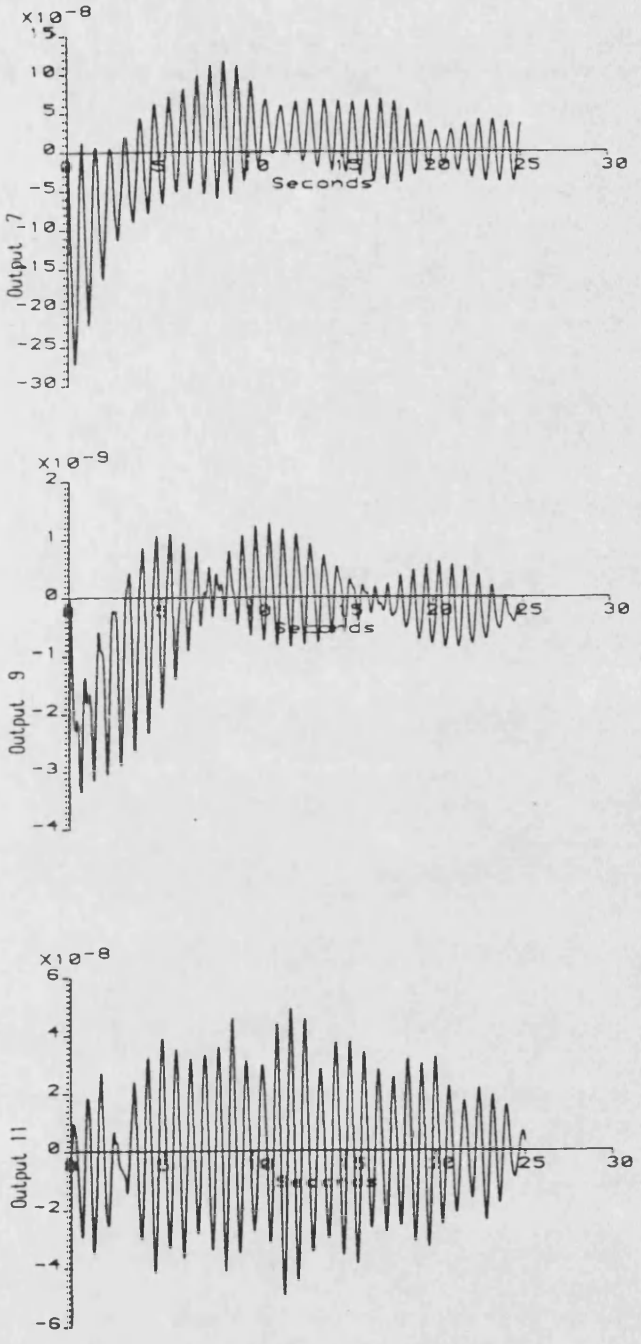


Figure 9.29: Payload responses of input 1 of Platform attitude model with initial baseline controller

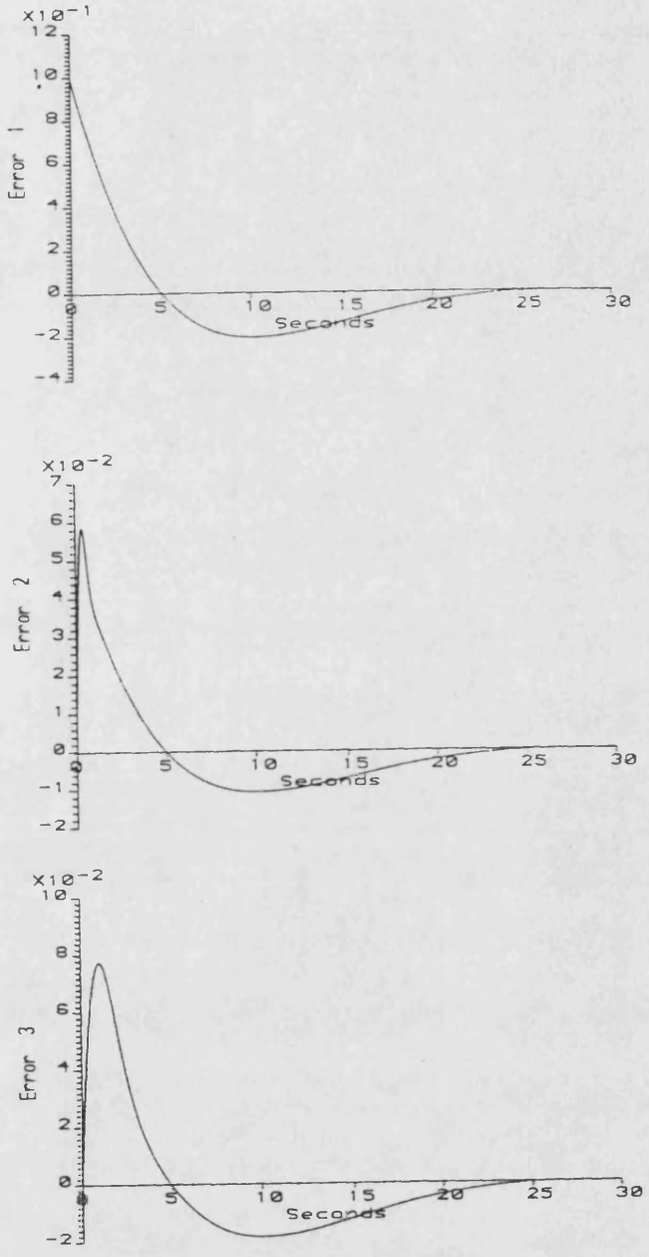


Figure 9.30: Error responses of input 1 of Platform attitude model with initial baseline controller

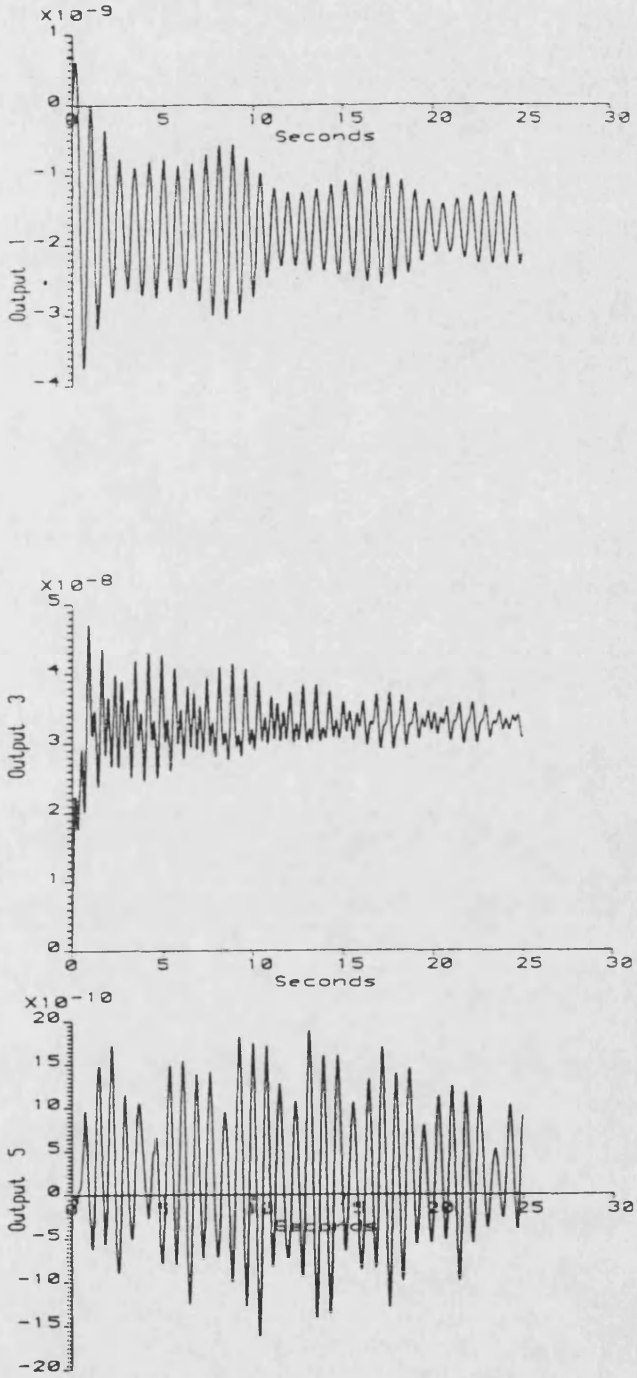


Figure 9.31: Attitude responses of input 2 of Platform attitude model with initial baseline controller

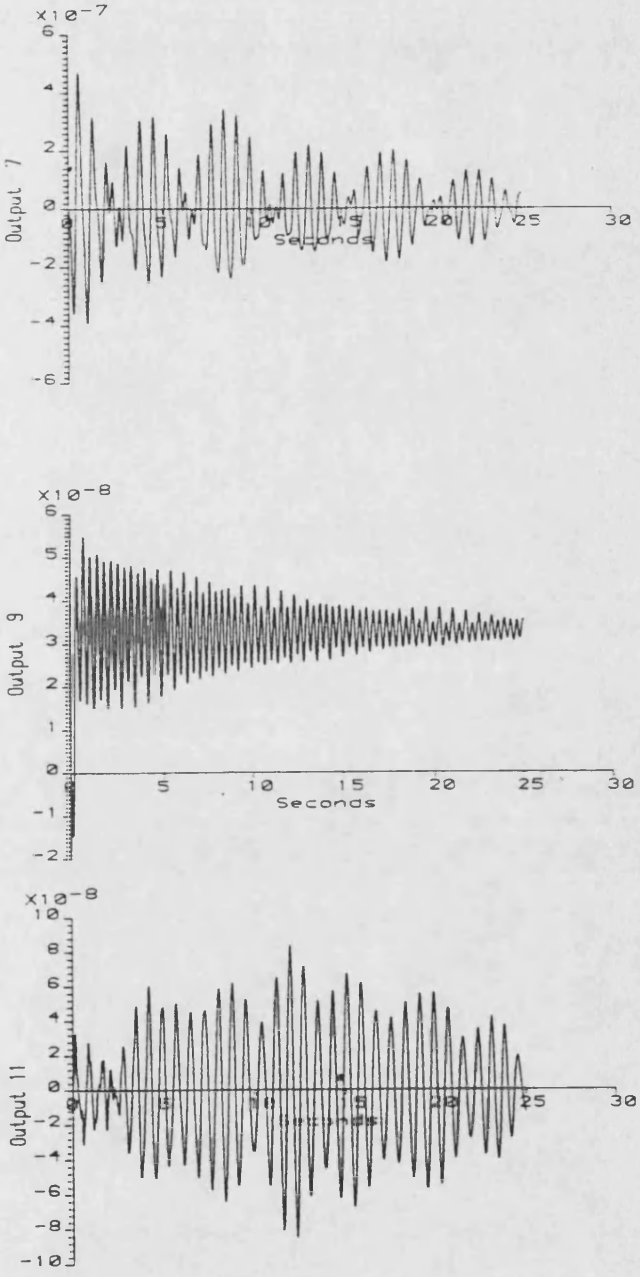


Figure 9.32: Payload responses of input 2 of Platform attitude model with initial baseline controller

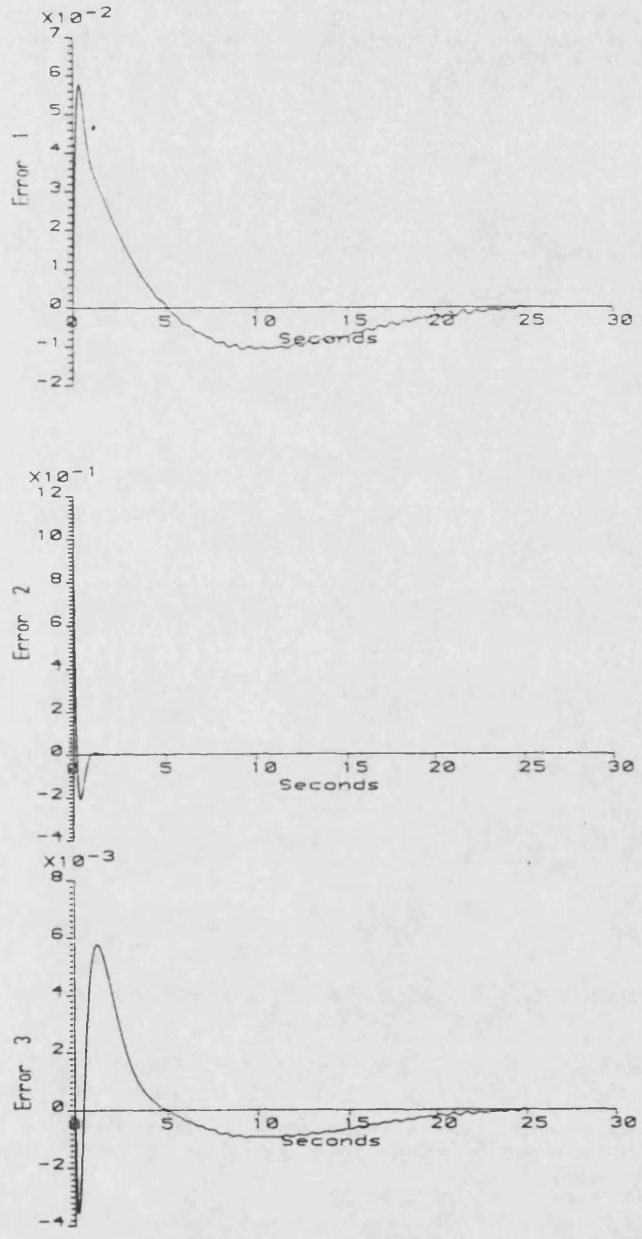


Figure 9.33: Error responses of input 2 of Platform attitude model with initial baseline controller

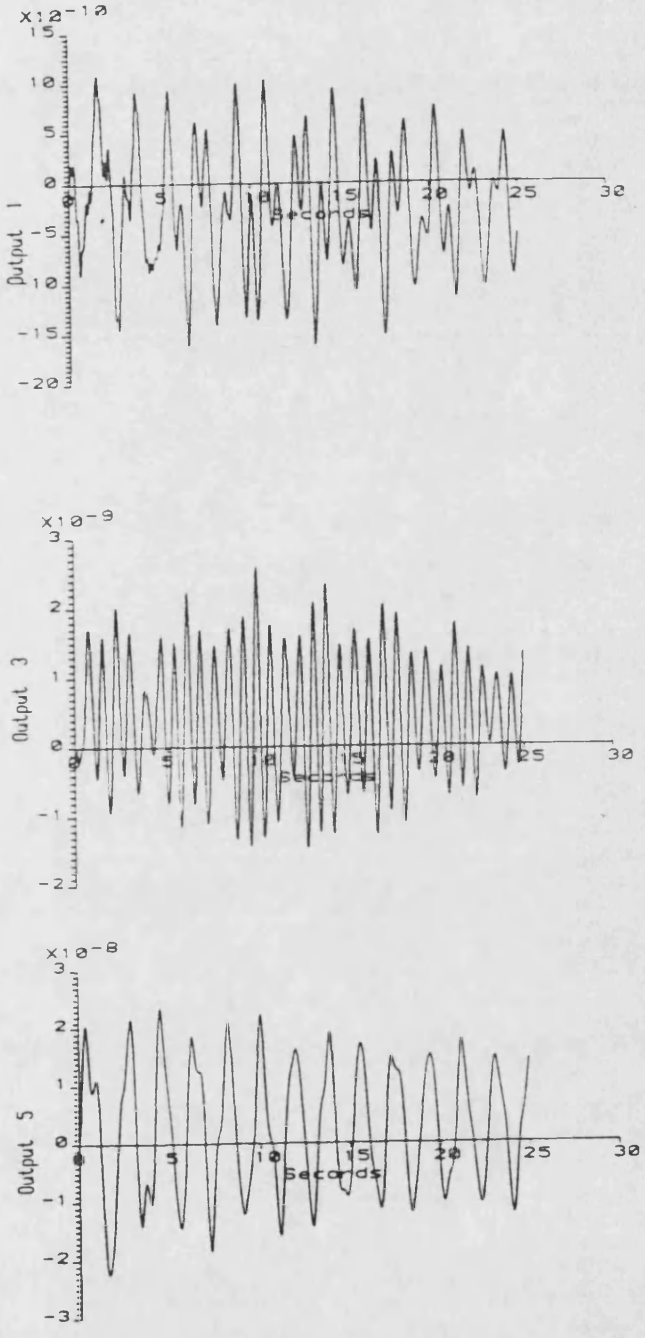


Figure 9.34: Attitude responses of input 3 of Platform attitude model with initial baseline controller

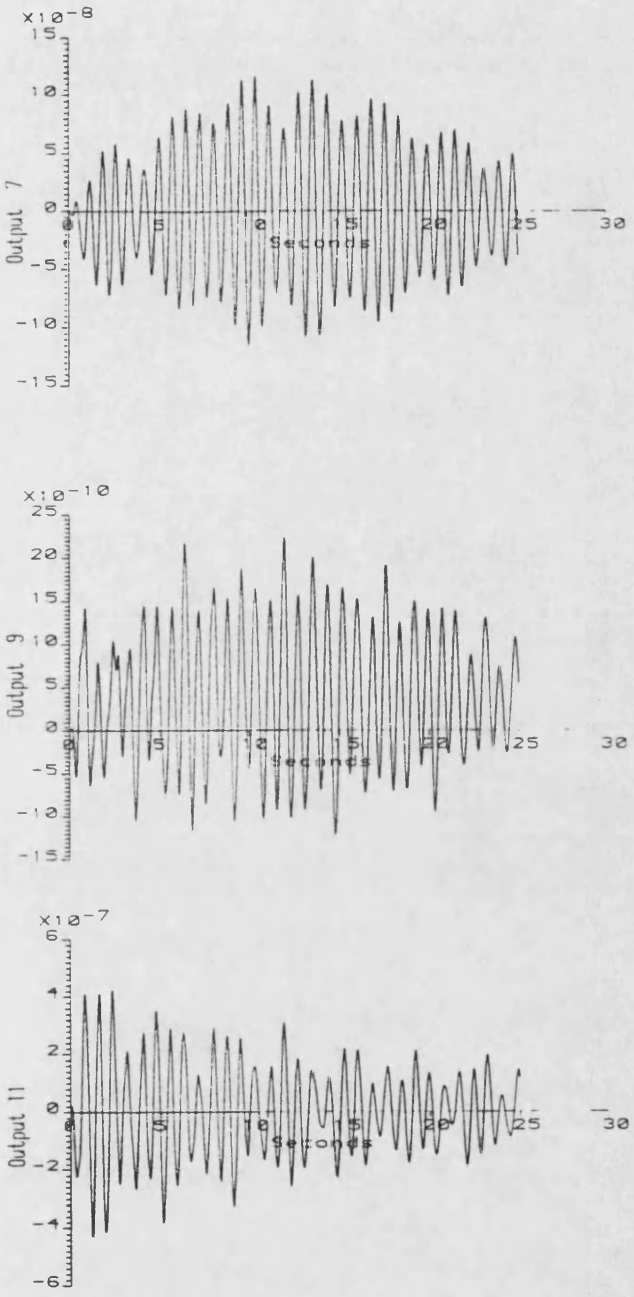


Figure 9.35: Payload responses of input 3 of Platform attitude model with initial baseline controller

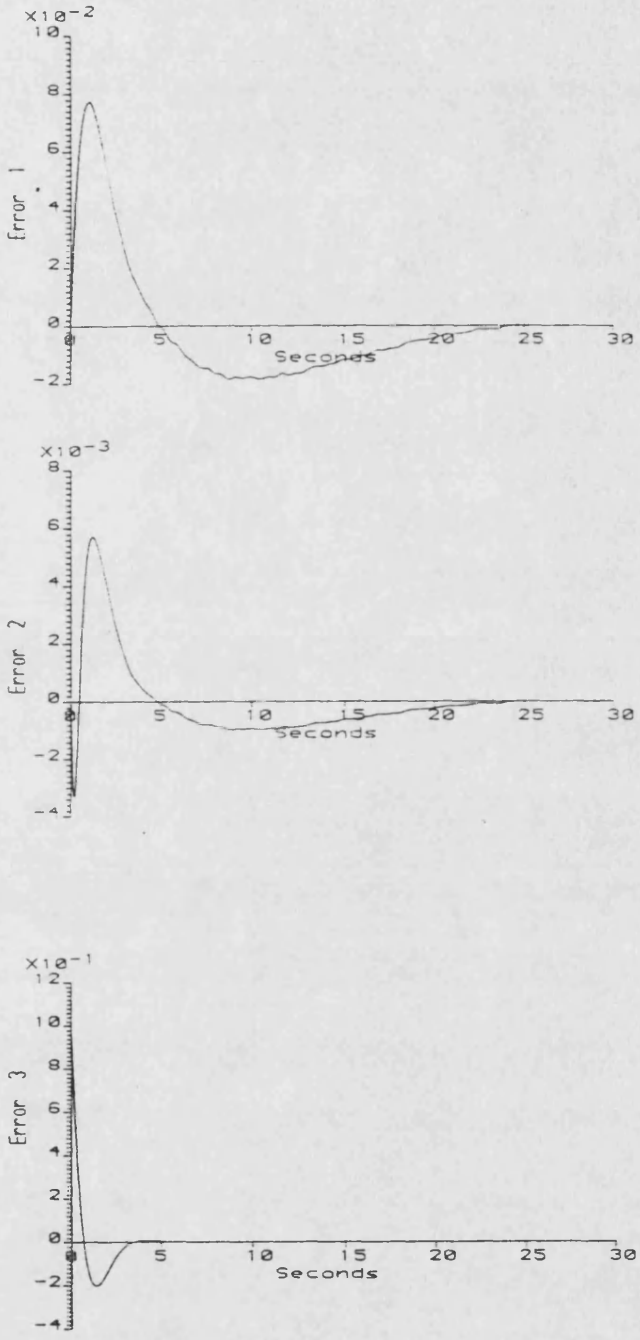


Figure 9.36: Error responses of input 3 of Platform attitude model with initial baseline controller

done by repeating the design process with increased weighting on the states corresponding to the elastic modes that require damping augmentation.

Similarly, if the oscillations present in the responses of node 26 cause problems, then the controller could be redesigned with increased damping and/or a frequency change of the offending modes. In the absence of any specific engineering requirements for the platform, the performance with this controller will be assumed to be acceptable.

However, examination of the state estimator poles in table 9.13 shows that certain of the poles are very much faster than the majority of the poles. These high frequency poles are associated with high gains, and such high gains are generally associated with a sensitivity to noise, and thus an attempt was made to redesign the state estimator such that the range of pole locations was reduced, but with the constraint that the closed loop performance was altered as little as possible. This was done by slightly reducing the weighting terms associated with the poles in the extreme positions. The poles of the redesigned state estimator are shown in table 9.14. Note that the state feedback matrix is unchanged.

The closed loop system formed by the platform attitude model and the controller based on this modified state estimator was simulated. The results of the three test cases as used above are shown in figure 9.37, through to figure 9.45. It can be seen that no noticeable degradation in performance has occurred.

9.4.2 Model Order Reduction—Controller Design

In a similar manner to that used for the earlier exercise, a modal cost analysis of the platform attitude model was carried out using the program MCA. The results of the modal cost analysis are shown in table 9.15 and figure 9.46. Model reduction was carried out using mode deletion of the modes with the lowest cost.

A series of reduced order models was considered, starting with a model comprised of the rigid modes plus the first ten elastic modes with the highest cost, which corresponds to a 26th order model with a model error of 7.6%. A controller was designed for this model using the program LQGDES, with similar weighting terms as used for the baseline controller. The poles of the resulting closed loop system are shown in table 9.16.

The order of the design model was further reduced by deleting the next two elastic modes with lowest cost, which resulted in a model error of 11.8%, and the design process repeated. The poles of the resulting closed loop

Designed Eigenvalues Of Plant		Designed Eigenvalues Of Estimator	
Real	Imaginary	Real	Imaginary
-0.31205	62.412	-0.31219	62.412
-0.31205	-62.412	-0.31219	-62.412
-0.28005	56.006	-0.28112	56.006
-0.28005	-56.006	-0.28112	-56.006
-0.19620	39.245	-0.19634	39.245
-0.19620	-39.245	-0.19634	-39.245
-0.18620	37.235	-0.18642	37.235
-0.18620	-37.235	-0.18642	-37.235
-0.15470	30.939	-0.15545	30.939
-0.15470	-30.939	-0.15545	-30.939
-0.14870	29.742	-0.14874	29.742
-0.14870	-29.742	-0.14874	-29.742
-0.14745	29.486	-0.14745	29.486
-0.14745	-29.486	-0.14745	-29.486
-0.13250	26.502	-0.13252	26.502
-0.13250	-26.502	-0.13252	-26.502
-0.12410	24.818	-0.12421	24.818
-0.12410	-24.818	-0.12421	-24.818
-0.10390	20.780	-10.466	0.49943
-0.10390	-20.780	-10.466	-0.49943
-0.86703E-01	17.341	-0.10398	20.780
-0.86703E-01	-17.341	-0.10398	-20.780
-4.0655	4.0656	-0.87375E-01	17.341
-4.0655	-4.0656	-0.87375E-01	-17.341
-0.50800E-01	10.159	-0.93736	0.44116
-0.50800E-01	-10.159	-0.93736	-0.44116
-0.48201E-01	9.6394	-0.88546E-01	0.87190E-01
-0.48201E-01	-9.6394	-0.88546E-01	-0.87190E-01
-0.47301E-01	9.4591	-0.16819E-01	3.3597
-0.47301E-01	-9.4591	-0.16819E-01	-3.3597
-0.46250E-01	9.2507	-0.15300E-01	3.0625
-0.46250E-01	-9.2507	-0.15300E-01	-3.0625
-0.43950E-01	8.7887	-0.39210E-01	7.8328
-0.43950E-01	-8.7887	-0.39210E-01	-7.8328
-0.40151E-01	8.0316	-0.40186E-01	8.0316
-0.40151E-01	-8.0316	-0.40186E-01	-8.0316
-0.39151E-01	7.8328	-0.50808E-01	10.159
-0.39151E-01	-7.8328	-0.50808E-01	-10.159
-1.1435	1.1436	-0.43960E-01	8.7887
-1.1435	-1.1436	-0.43960E-01	-8.7887
-0.16802E-01	3.3597	-0.46257E-01	9.2507
-0.16802E-01	-3.3597	-0.46257E-01	-9.2507
-0.15300E-01	3.0625	-0.47338E-01	9.4591
-0.15300E-01	-3.0625	-0.47338E-01	-9.4591
-0.15622	0.15631	-0.48274E-01	9.6394
-0.15622	-0.15631	-0.48274E-01	-9.6394

Table 9.14: Closed loop poles of modified baseline state estimator

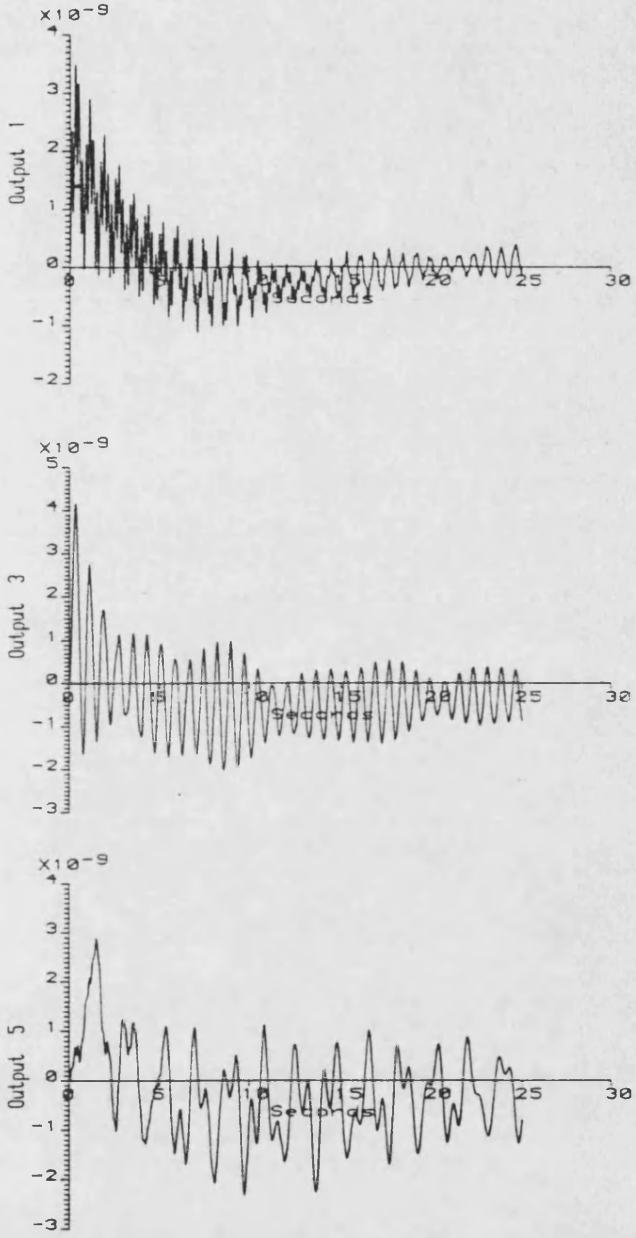


Figure 9.37: Attitude responses of input 1 of Platform attitude model with modified baseline controller

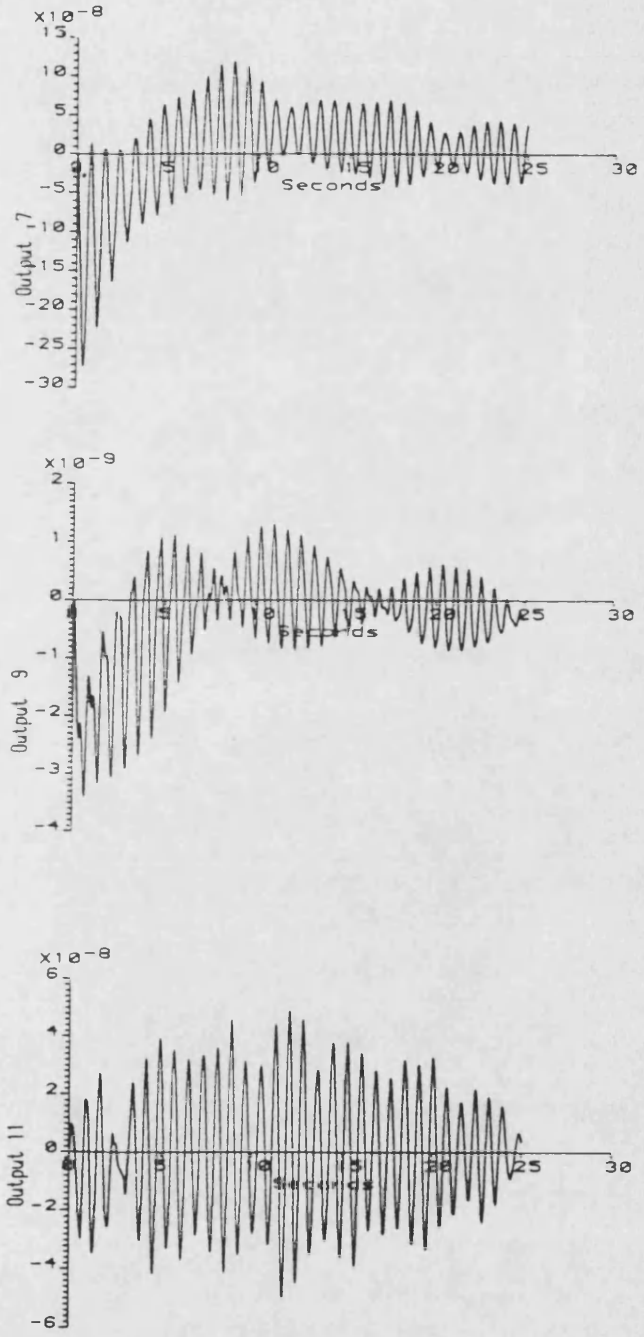


Figure 9.38: Payload responses of input 1 of Platform attitude model with modified baseline controller

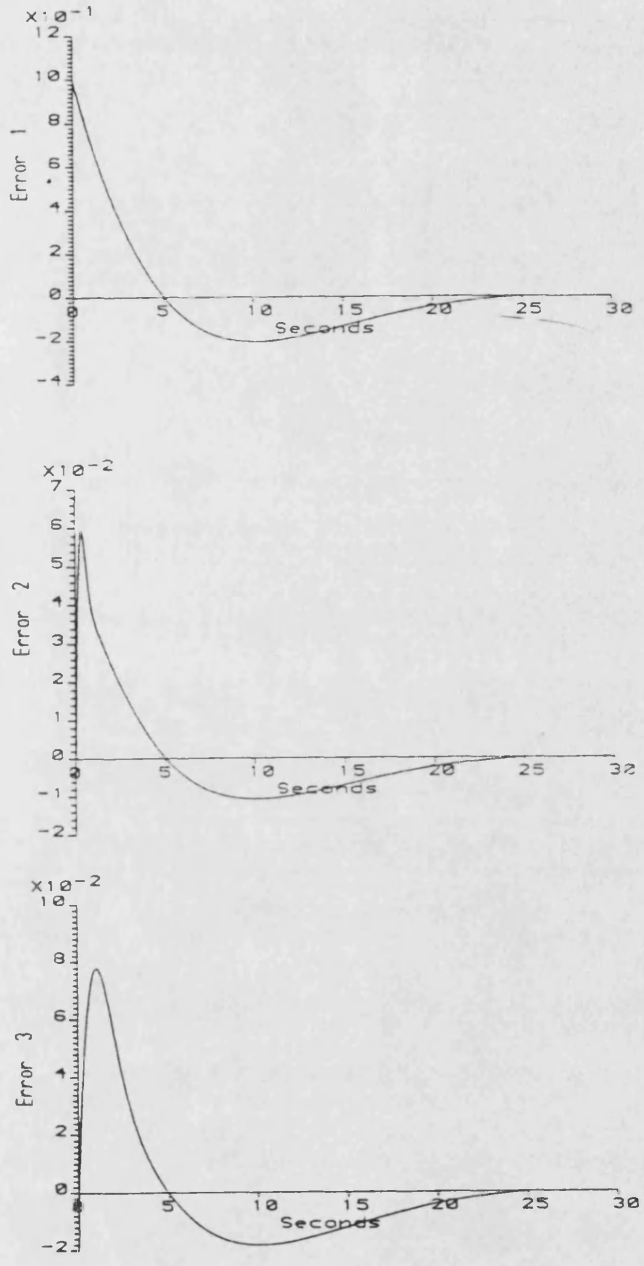


Figure 9.39: Error responses of input 1 of Platform attitude model with modified baseline controller

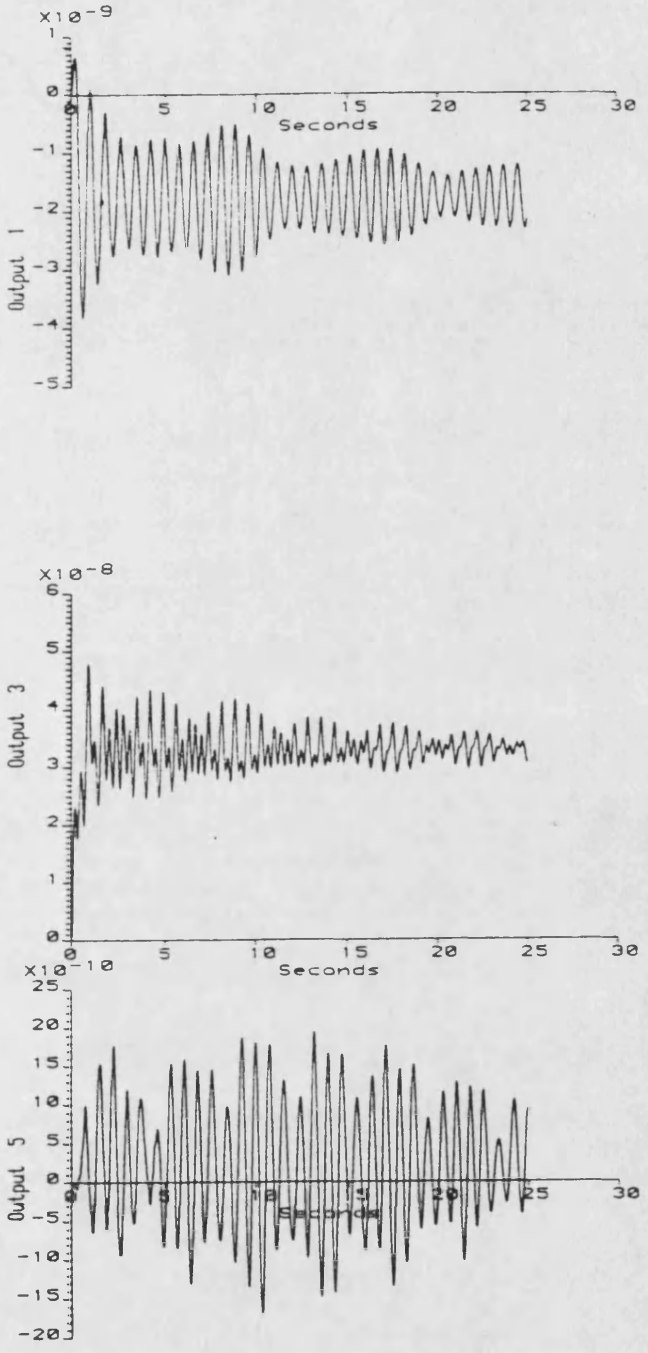


Figure 9.40: Attitude responses of input 2 of Platform attitude model with modified baseline controller

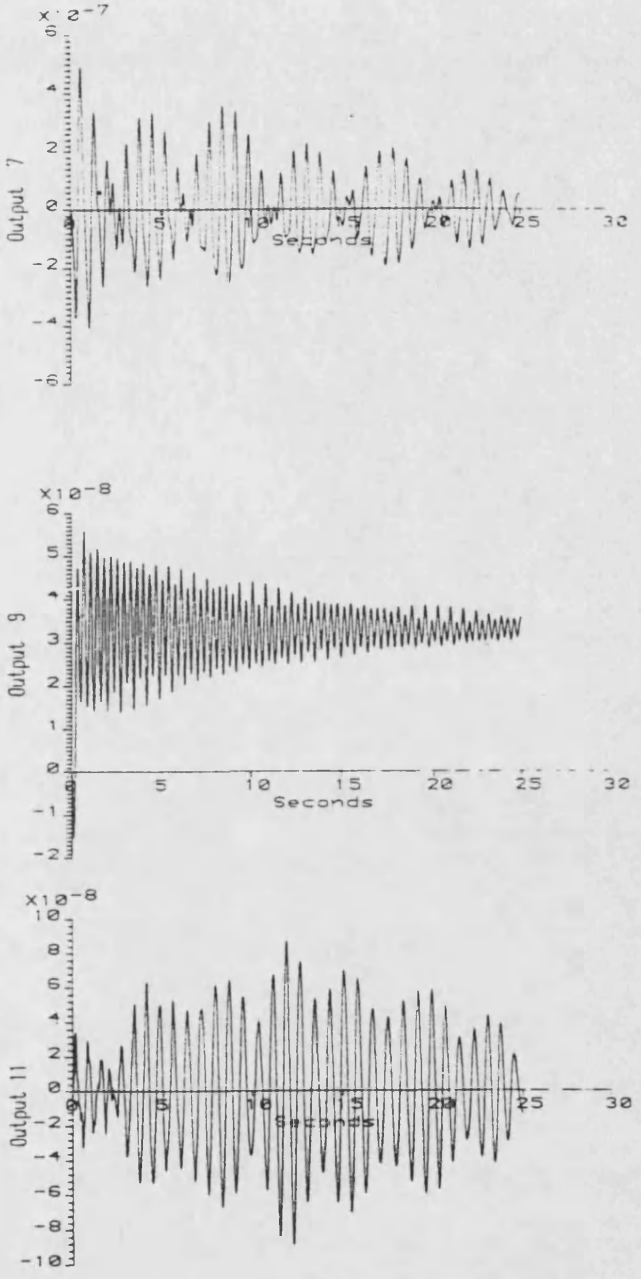


Figure 9.41: Payload responses of input 2 of Platform attitude model with modified baseline controller

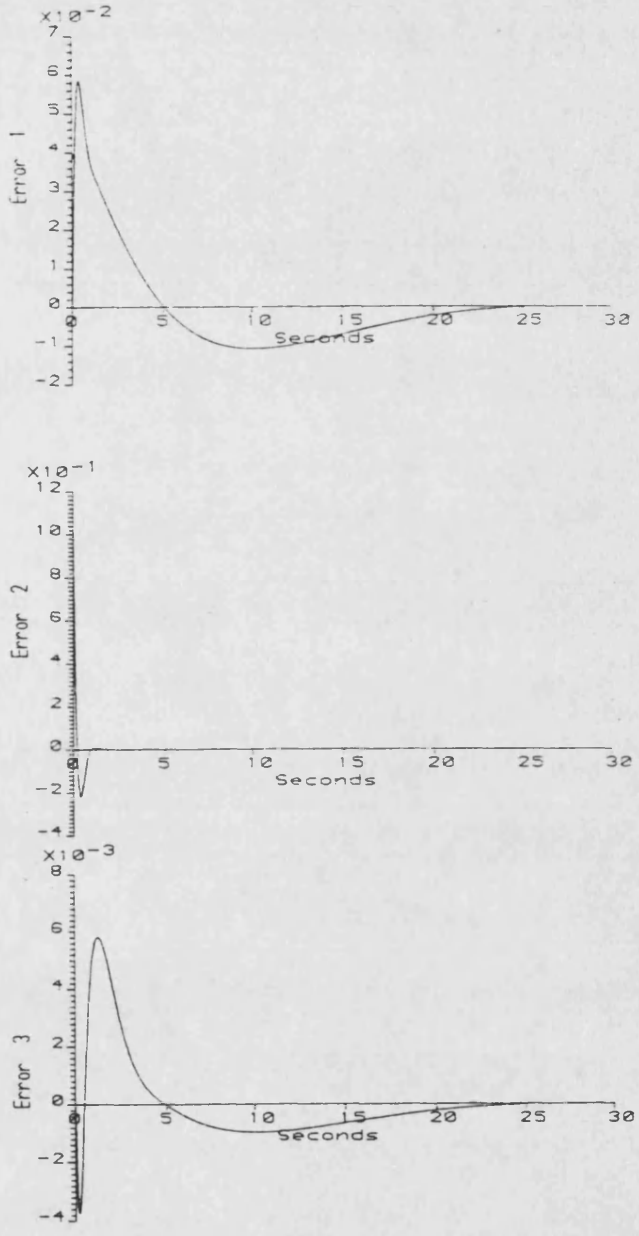


Figure 9.42: Error responses of input 2 of Platform attitude model with modified baseline controller

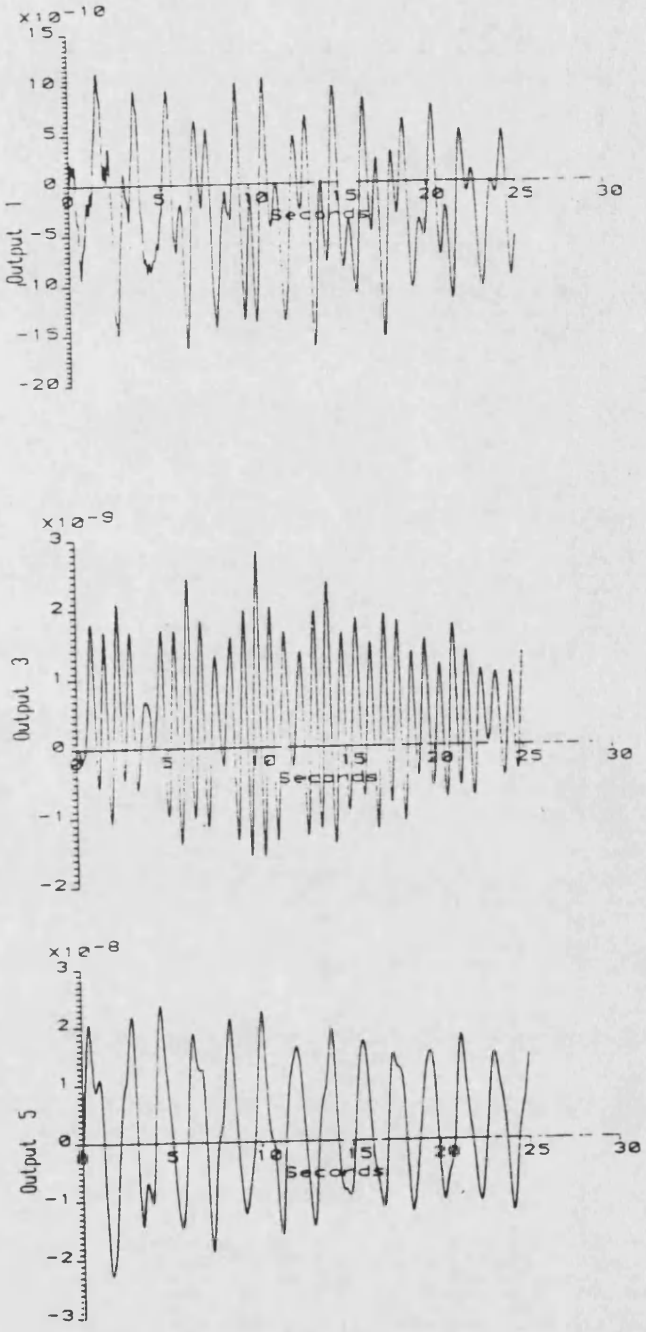


Figure 9.43: Attitude responses of input 3 of Platform attitude model with modified baseline controller

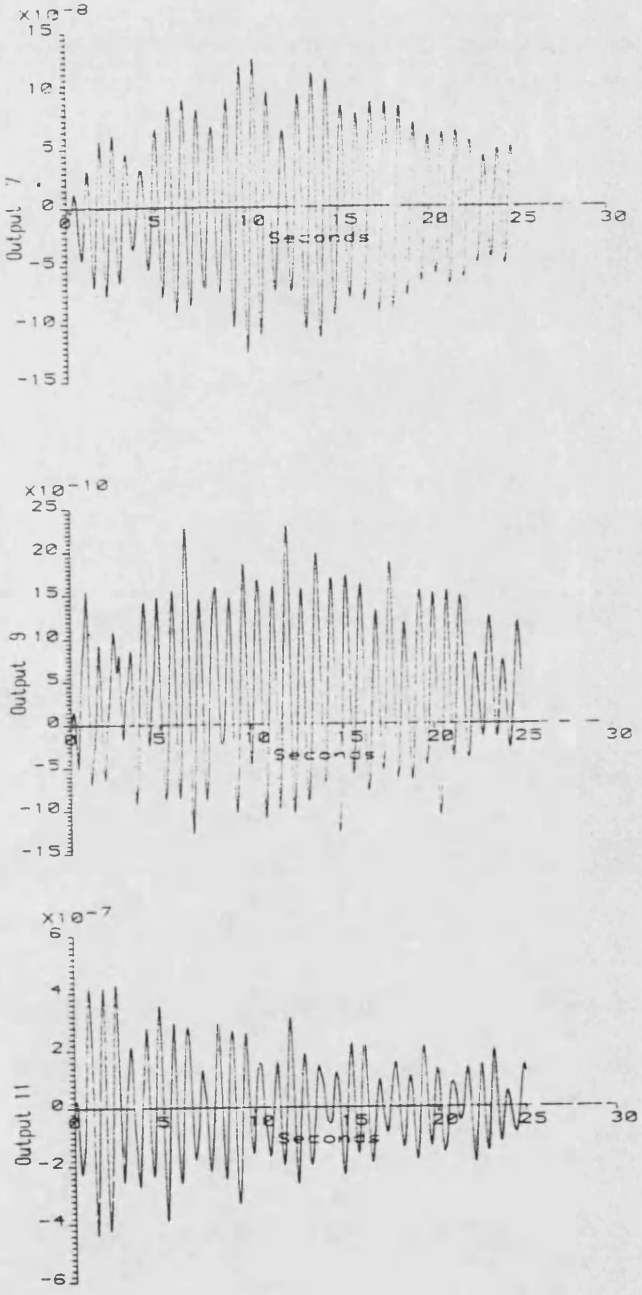


Figure 9.44: Payload responses of input 3 of Platform attitude model with modified baseline controller

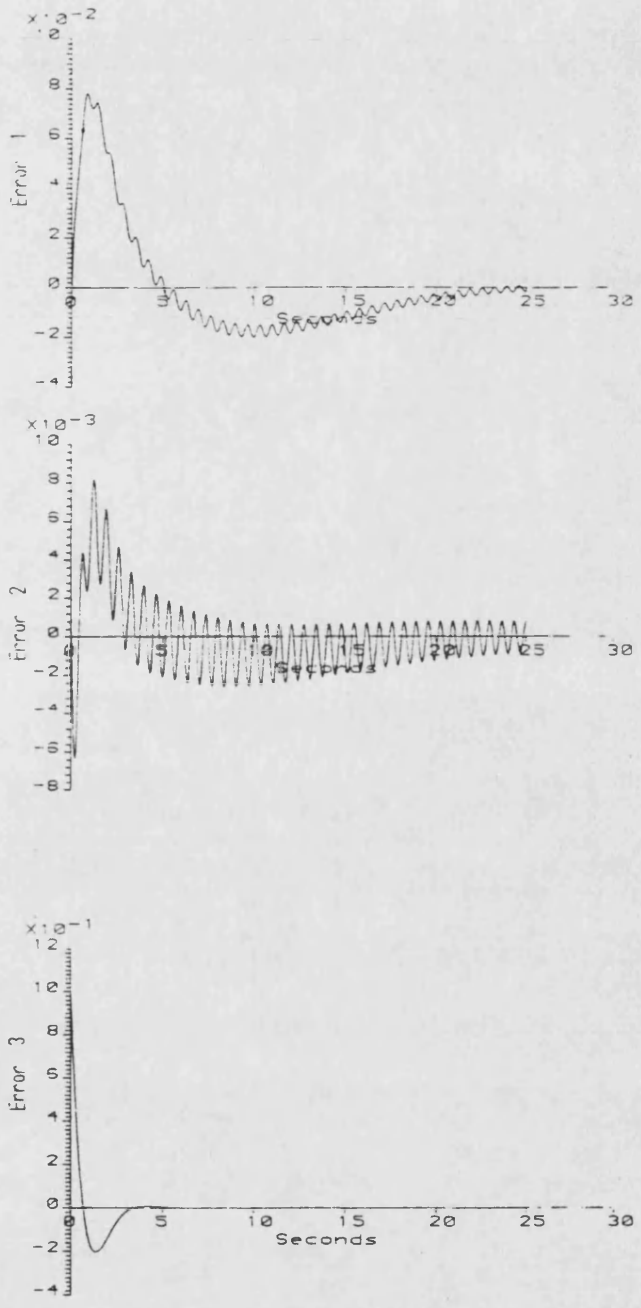


Figure 9.45: Error responses of input 3 of Platform attitude model with modified baseline controller

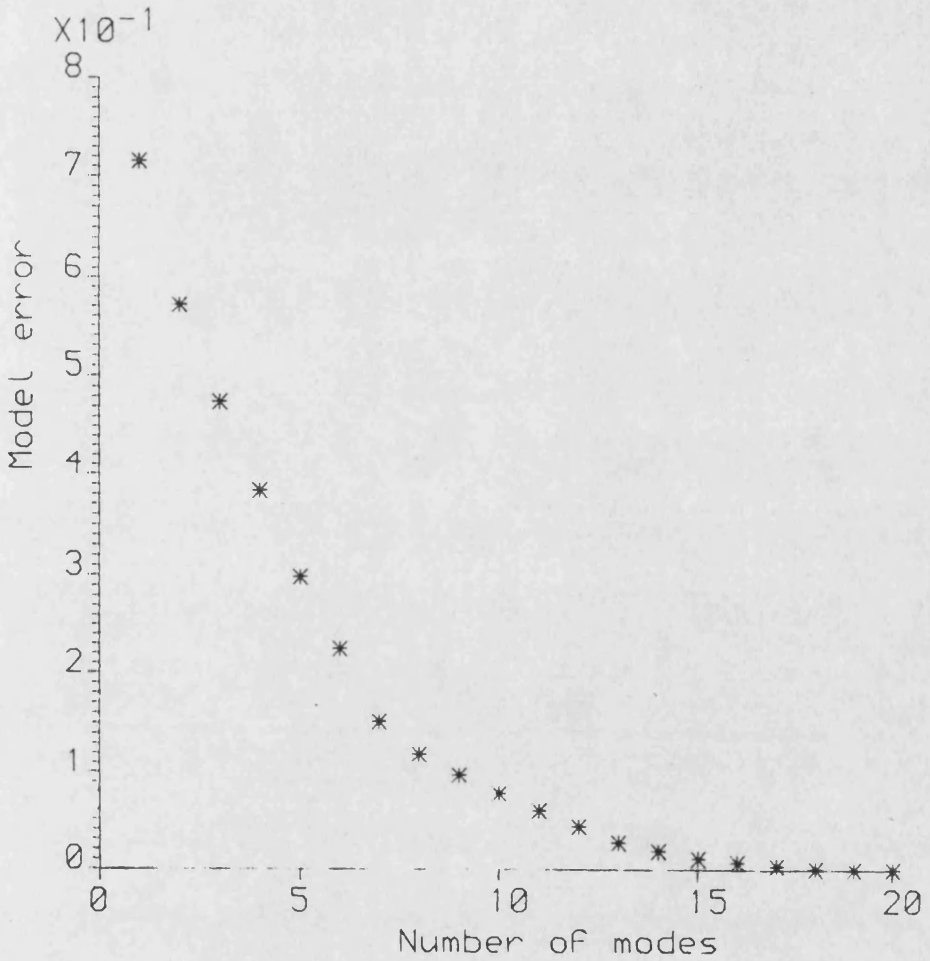


Figure 9.46: Model error function for Platform attitude model

Mode no.	Modal Cost	Mode no.	Modal Cost
13	0.17271×10^{-4}	14	0.10808×10^{-5}
5	0.87424×10^{-5}	9	0.99617×10^{-6}
7	0.58525×10^{-5}	20	0.96735×10^{-6}
6	0.55493×10^{-5}	21	0.50949×10^{-6}
10	0.51440×10^{-5}	12	0.46750×10^{-6}
11	0.45010×10^{-5}	18	0.22776×10^{-6}
19	0.44423×10^{-5}	23	0.20510×10^{-6}
22	0.19998×10^{-5}	16	0.16741×10^{-6}
8	0.13500×10^{-5}	17	0.11814×10^{-7}
15	0.11812×10^{-5}	4	0.56855×10^{-9}

Table 9.15: Modal cost analysis of Platform attitude model

system are shown in table 9.17.

This procedure of further design model order reduction and subsequent controller design was repeated up to, and including, the model comprised of the rigid modes alone. The poles of the resulting closed loop system are shown for the 18th order case (a model error of 22.4%) in table 9.18. Note that the closed loop system is unstable, and this is also the case for all the designs based on models of lower order than this.

Designed Eigenvalues Of Plant

Real	Imaginary
-0.28005	56.006
-0.28005	-56.006
-0.15470	30.939
-0.15470	-30.939
-0.12410	24.818
-0.12410	-24.818
-0.86703E-01	17.341
-0.86703E-01	-17.341
-4.0655	4.0656
-4.0655	-4.0656
-0.48201E-01	9.6394
-0.48201E-01	-9.6394
-0.47301E-01	9.4591
-0.47301E-01	-9.4591
-0.43950E-01	8.7887
-0.43950E-01	-8.7887
-0.39151E-01	7.8328
-0.39151E-01	-7.8328
-0.40151E-01	8.0316
-0.40151E-01	-8.0316
-0.16802E-01	3.3597
-0.16802E-01	-3.3597
-1.1435	1.1436
-1.1435	-1.1436
-0.15625	0.15121
-0.15625	-0.15121

Designed Eigenvalues Of Estimator

Real	Imaginary
-0.28112	56.006
-0.28112	-56.006
-0.15545	30.939
-0.15545	-30.939
-0.12421	24.818
-0.12421	-24.818
-0.87375E-01	17.341
-0.87375E-01	-17.341
-10.466	0.49943
-10.466	-0.49943
-0.93736	0.44116
-0.93736	-0.44116
-0.88536E-01	0.87180E-01
-0.88536E-01	-0.87180E-01
-0.16819E-01	3.3597
-0.16819E-01	-3.3597
-0.48274E-01	9.6394
-0.48274E-01	-9.6394
-0.47338E-01	9.4591
-0.47338E-01	-9.4591
-0.43960E-01	8.7887
-0.43960E-01	-8.7887
-0.40186E-01	8.0316
-0.40186E-01	-8.0316
-0.39210E-01	7.8328
-0.39210E-01	-7.8328

Eigenvalues Of Closed Loop System

Real	Imaginary
-0.29355	62.764
-0.29355	-62.764
-0.28005	56.006
-0.28005	-56.006
-0.28114	56.006
-0.28114	-56.006
-0.17607	39.948
-0.17607	-39.948
-0.27908	37.621
-0.27908	-37.621
-0.14871	30.105
-0.14871	-30.105
-0.14879	29.491
-0.14879	-29.491
-0.15470	30.939
-0.15470	-30.939
-0.15548	30.939
-0.15548	-30.939
-0.12553	26.748
-0.12553	-26.748
-0.14920	22.162
-0.14920	-22.162
-0.12410	24.818
-0.12410	-24.818
-0.12421	24.818
-0.12421	-24.818
-11.375	0.00000E+00
-0.86703E-01	17.341
-0.86703E-01	-17.341
-0.87377E-01	17.341
-0.87377E-01	-17.341
-8.9158	0.00000E+00
-4.0245	4.3482
-4.0245	-4.3482
-0.62206E-01	10.650
-0.62206E-01	-10.650
-0.70894	0.99486
-0.70894	-0.99486
-1.2867	0.00000E+00
-1.1138	0.00000E+00
-0.29622E-01	0.91114E-01
-0.29622E-01	-0.91114E-01
-0.10130	0.00000E+00
-0.74336	0.00000E+00
-0.15523E-01	3.0630
-0.15523E-01	-3.0630
-0.25724	9.2987
-0.25724	-9.2987
-0.48201E-01	9.6394
-0.48201E-01	-9.6394
-0.48229E-01	9.6394
-0.48229E-01	-9.6394
-0.47301E-01	9.4591
-0.47301E-01	-9.4591
-0.47318E-01	9.4590
-0.47318E-01	-9.4590
-0.43950E-01	8.7887
-0.43950E-01	-8.7887
-0.43959E-01	8.7887
-0.43959E-01	-8.7887
-0.40151E-01	8.0316
-0.40151E-01	-8.0316
-0.40183E-01	8.0317
-0.40183E-01	-8.0317
-0.39151E-01	7.8328
-0.39151E-01	-7.8328
-0.39189E-01	7.8328
-0.39189E-01	-7.8328
-0.16802E-01	3.3597
-0.16802E-01	-3.3597
-0.16814E-01	3.3597
-0.16814E-01	-3.3597

Table 9.16: Closed loop poles of Platform attitude model with 26th order controller

Designed Eigenvalues Of Plant

Real	Imaginary
-0.28005	56.006
-0.28005	-56.006
-0.15470	30.939
-0.15470	-30.939
-0.86703E-01	17.341
-0.86703E-01	-17.341
-4.0655	4.0654
-4.0655	-4.0654
-0.48201E-01	9.6394
-0.48201E-01	-9.6394
-0.47301E-01	9.4591
-0.47301E-01	-9.4591
-0.40151E-01	8.0316
-0.40151E-01	-8.0316
-0.39151E-01	7.8328
-0.39151E-01	-7.8328
-1.1435	1.1432
-1.1435	-1.1432
-0.16802E-01	3.3597
-0.16802E-01	-3.3597
-0.15662	0.17168
-0.15662	-0.17168

Designed Eigenvalues Of Estimator

Real	Imaginary
-0.28112	56.006
-0.28112	-56.006
-0.15545	30.939
-0.15545	-30.939
-0.87375E-01	17.341
-0.87375E-01	-17.341
-10.466	0.49943
-10.466	-0.49943
-0.93737	0.44116
-0.93737	-0.44116
-0.88436E-01	0.87204E-01
-0.88436E-01	-0.87204E-01
-0.48274E-01	9.6394
-0.48274E-01	-9.6394
-0.47338E-01	9.4591
-0.47338E-01	-9.4591
-0.40186E-01	8.0316
-0.40186E-01	-8.0316
-0.39210E-01	7.8328
-0.39210E-01	-7.8328
-0.16819E-01	3.3597
-0.16819E-01	-3.3597

Eigenvalues Of Closed Loop System

Real	Imaginary
-0.28821	62.791
-0.28821	-62.791
-0.28005	56.006
-0.28005	-56.006
-0.28116	56.006
-0.28116	-56.006
-0.15881	39.962
-0.15881	-39.962
-0.28896	37.752
-0.28896	-37.752
-0.14404	30.121
-0.14404	-30.121
-0.14893	29.490
-0.14893	-29.490
-0.15470	30.939
-0.15470	-30.939
-0.15550	30.939
-0.15550	-30.939
-0.11973	26.761
-0.11973	-26.761
-0.31107	25.178
-0.31107	-25.178
-0.14595	22.189
-0.14595	-22.189
-11.705	0.00000E+00
-0.86703E-01	17.341
-0.86703E-01	-17.341
-0.87365E-01	17.341
-0.87365E-01	-17.341
-7.9336	0.00000E+00
-3.8171	4.3302
-3.8171	-4.3302
-0.15841E-01	10.739
-0.15841E-01	-10.739
-2.0608	0.00000E+00
-0.70196	0.99247
-0.70196	-0.99247
-1.1207	0.00000E+00
-1.2842	0.00000E+00
-0.10281E-01	0.65212E-01
-0.10281E-01	-0.65212E-01
-0.74869E-01	0.00000E+00
-0.15524E-01	3.0630
-0.15524E-01	-3.0630
-0.93229E-01	9.9380
-0.93229E-01	-9.9380
-0.49312E-01	9.3068
-0.49312E-01	-9.3068
-0.48201E-01	9.6394
-0.48201E-01	-9.6394
-0.48227E-01	9.6394
-0.48227E-01	-9.6394
-0.47316E-01	9.4591
-0.47316E-01	-9.4591
-0.47300E-01	9.4591
-0.47300E-01	-9.4591
-0.40151E-01	8.0316
-0.40151E-01	-8.0316
-0.40166E-01	8.0317
-0.40166E-01	-8.0317
-0.39151E-01	7.8328
-0.39151E-01	-7.8328
-0.39188E-01	7.8328
-0.39188E-01	-7.8328
-0.16802E-01	3.3597
-0.16802E-01	-3.3597
-0.16814E-01	3.3597
-0.16814E-01	-3.3597

Table 9.17: Closed loop poles of Platform attitude model with 22nd order controller

Designed Eigenvalues Of Plant

Real	Imaginary
-0.86703E-01	17.341
-0.86703E-01	-17.341
-4.0655	4.0656
-4.0655	-4.0656
-0.47301E-01	9.4591
-0.47301E-01	-9.4591
-0.48201E-01	9.6394
-0.48201E-01	-9.6394
-0.40151E-01	8.0316
-0.40151E-01	-8.0316
-0.39151E-01	7.8328
-0.39151E-01	-7.8328
-1.1435	1.1434
-1.1435	-1.1434
-0.16802E-01	3.3597
-0.16802E-01	-3.3597
-0.15632	0.16289
-0.15632	-0.16289

Designed Eigenvalues Of Estimator

Real	Imaginary
-0.87375E-01	17.341
-0.87375E-01	-17.341
-10.466	0.49943
-10.466	-0.49943
-0.48274E-01	9.6394
-0.48274E-01	-9.6394
-0.47338E-01	9.4591
-0.47338E-01	-9.4591
-0.39210E-01	7.8328
-0.39210E-01	-7.8328
-0.40186E-01	8.0316
-0.40186E-01	-8.0316
-0.16819E-01	3.3597
-0.16819E-01	-3.3597
-0.93737	0.44116
-0.93737	-0.44116
-0.88501E-01	0.87153E-01
-0.88501E-01	-0.87153E-01

Eigenvalues Of Closed Loop System

Real	Imaginary
-0.27090	62.837
-0.27090	-62.837
-0.42182	56.771
-0.42182	-56.771
1.1720	41.765
1.1720	-41.765
0.12340E-01	39.150
0.12340E-01	-39.150
-0.19834	37.389
-0.19834	-37.389
-0.16476	30.057
-0.16476	-30.057
-0.14907	29.490
-0.14907	-29.490
-0.14179	26.711
-0.14179	-26.711
-0.30287	25.140
-0.30287	-25.140
-0.17588	22.107
-0.17588	-22.107
-11.801	0.00000E+00
-0.86703E-01	17.341
-0.86703E-01	-17.341
-0.87351E-01	17.341
-0.87351E-01	-17.341
-7.6893	0.00000E+00
-3.7680	4.5842
-3.7680	-4.5842
-4.2983	0.00000E+00
-0.10097	10.638
-0.10097	-10.638
-0.70136	0.99189
-0.70136	-0.99189
-1.3028	0.00000E+00
-1.1088	0.00000E+00
-0.16592E-02	0.38182E-01
-0.16592E-02	-0.38182E-01
-0.63857E-01	0.00000E+00
-0.15532E-01	3.0630
-0.15532E-01	-3.0630
-0.45228	9.5429
-0.45228	-9.5429
-0.41986E-03	9.1810
-0.41986E-03	-9.1810
-0.48201E-01	9.6394
-0.48201E-01	-9.6394
-0.48229E-01	9.6394
-0.48229E-01	-9.6394
-0.47292E-01	9.4590
-0.47292E-01	-9.4590
-0.47301E-01	9.4591
-0.47301E-01	-9.4591
-0.39151E-01	7.8328
-0.39151E-01	-7.8328
-0.39188E-01	7.8328
-0.39188E-01	-7.8328
-0.40151E-01	8.0316
-0.40151E-01	-8.0316
-0.40168E-01	8.0317
-0.40168E-01	-8.0317
-0.16802E-01	3.3597
-0.16802E-01	-3.3597
-0.16814E-01	3.3597
-0.16814E-01	-3.3597

Table 9.18: Closed loop poles of Platform attitude model with 18th order controller

Note that the effects of "spillover" (see Chapter 4) on the unmodelled modes gets more pronounced as the degree of model reduction increases. The system formed by the platform attitude model and the 26th order controller was simulated. Although the system based on the 22nd order controller is stable, some of the poles have much smaller stability margins than the open loop model, unlike the system with the 26th order controller, thus the latter controller was adopted for simulation. The responses for the three step demand cases are shown in figure 9.47 through to figure 9.55.

Note that the dynamic portions of the output responses are not noticeably different from those of the baseline controller. However, the steady state terms are different from the baseline case, and also the elastic modes are more in evidence in the error responses than in the baseline case. Reasons for this will be examined later in the chapter.

9.4.3 Controller Design—Controller Order Reduction

As in the previous exercise, the baseline controller design was used as the basis for an attempt to derive an "equivalent" reduced order controller. Again, the program QCOVER was used with a variety of initial conditions, and again the program was plagued with numerical difficulties and no useful results were obtained.

9.4.4 Output Feedback Controller

An output feedback based controller was designed for the platform attitude model using the program RMDES. The controller gains were selected such that the poles of the rigid modes were the same as the poles of the rigid modes of the system with the baseline controller (see Section 9.4.1). The poles of the resulting closed loop system are shown in table 9.19.

Examination of the poles of the closed loop system highlight a problem associated with the design of "high performance" output feedback based controllers for flexible spacecraft, that is the possibly excessive movements of some poles from their open loop locations. It can be seen from table 9.19 that although most of the elastic modes have moderate increases in stability (larger real components), some of the poles shifted a great deal, not all of them in the direction of increased stability. Note also the large range of values over which the poles stretch (of the order of 10^{12}), and because of this large degree of numerical stiffness, and the extremely slow pole pair which would dominate the response, the system was not simulated.

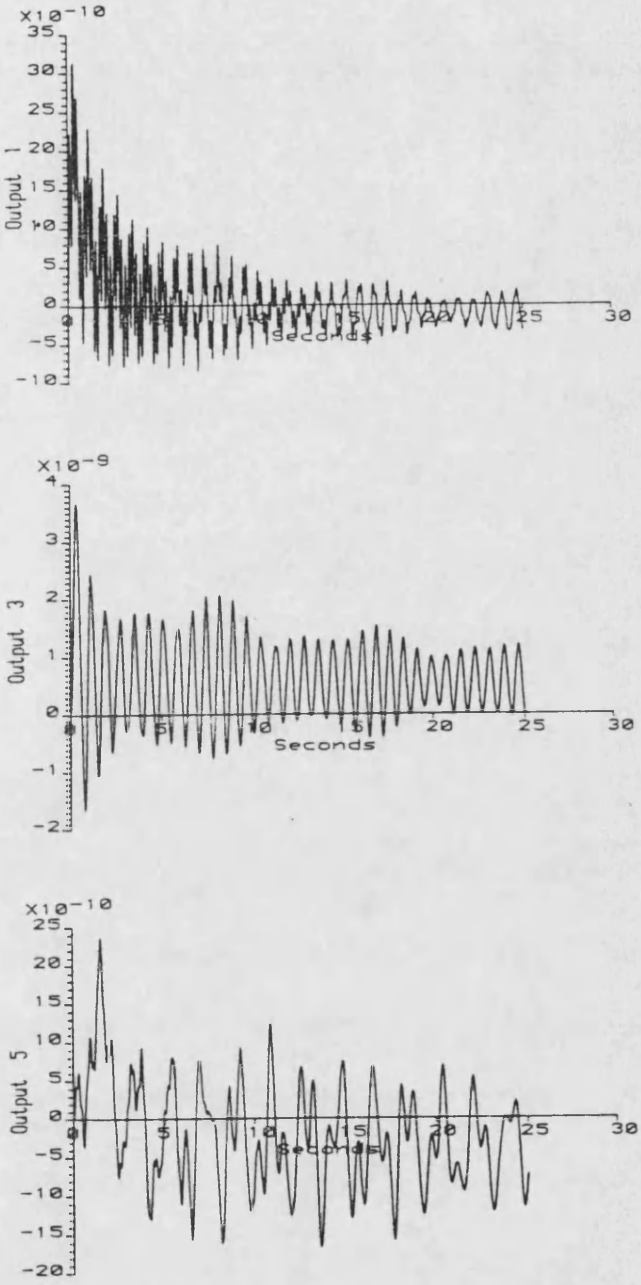


Figure 9.47: Attitude responses of input 1 of Platform attitude model with 26th order controller

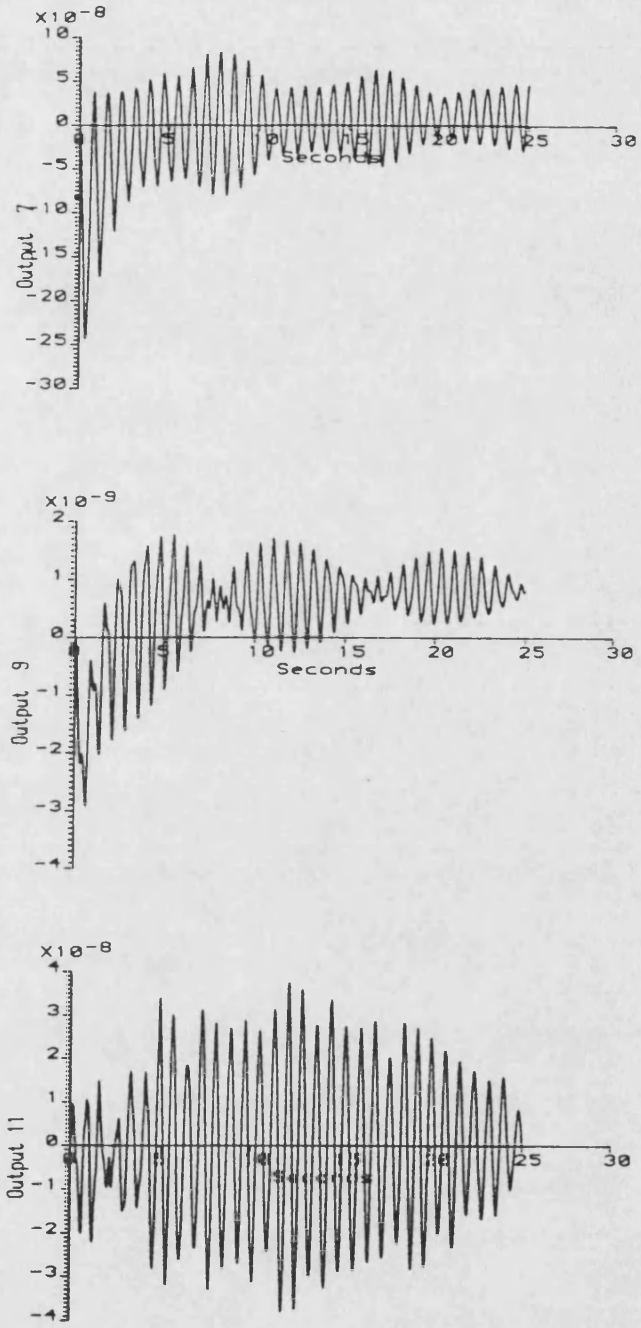


Figure 9.48: Payload responses of input 1 of Platform attitude model with 26th order controller

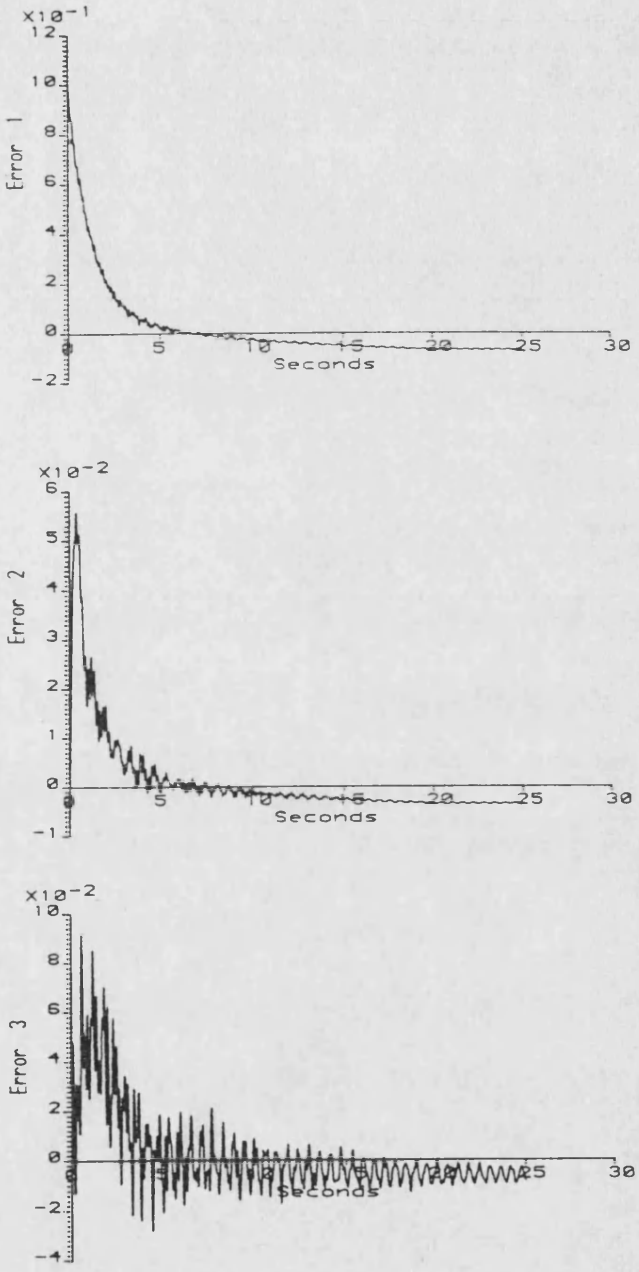


Figure 9.49: Error responses of input 1 of Platform attitude model with 26th order controller

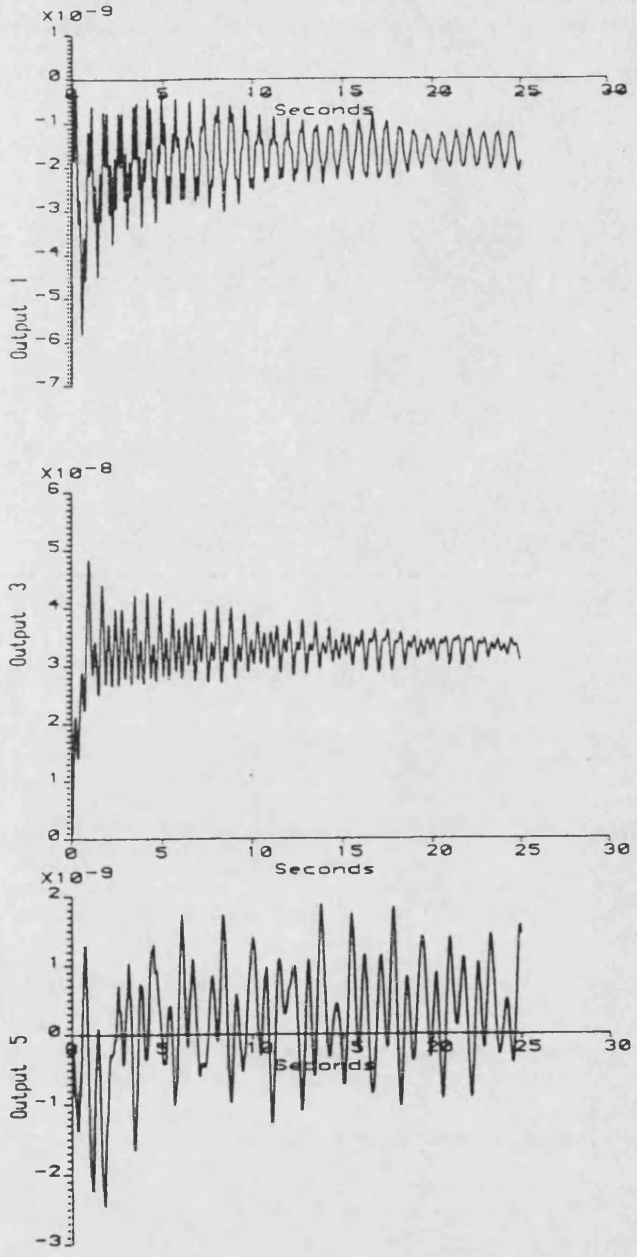


Figure 9.50: Attitude responses of input 2 of Platform attitude model with 26th order controller

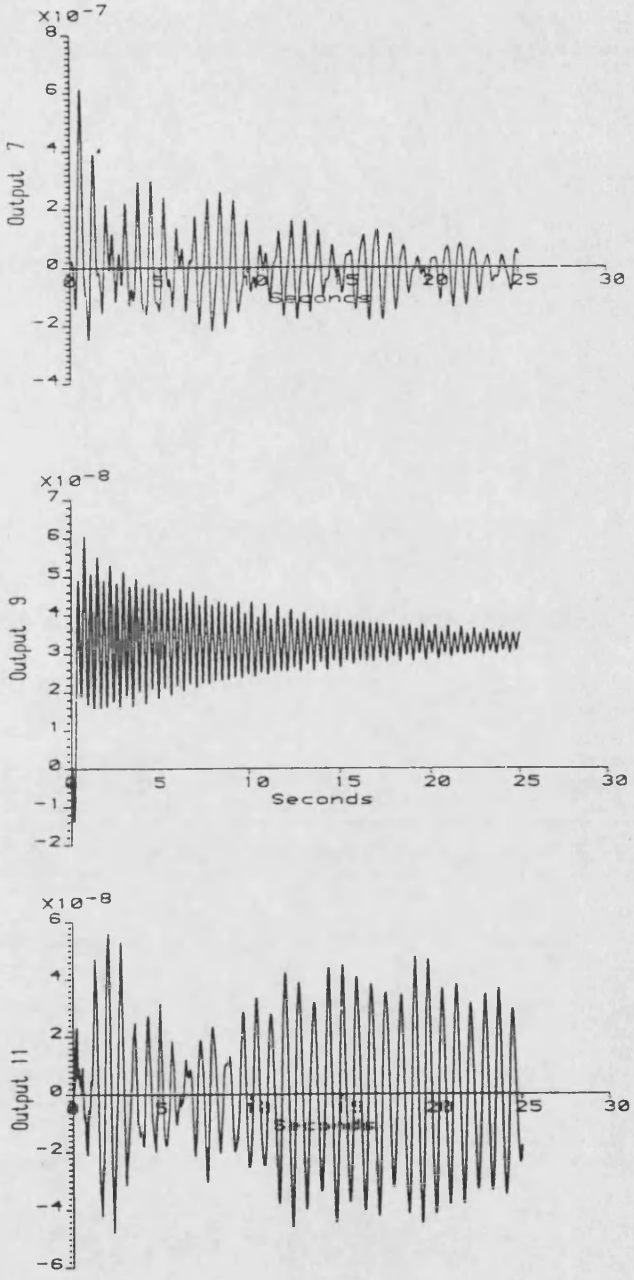


Figure 9.51: Payload responses of input 2 of Platform attitude model with 26th order controller

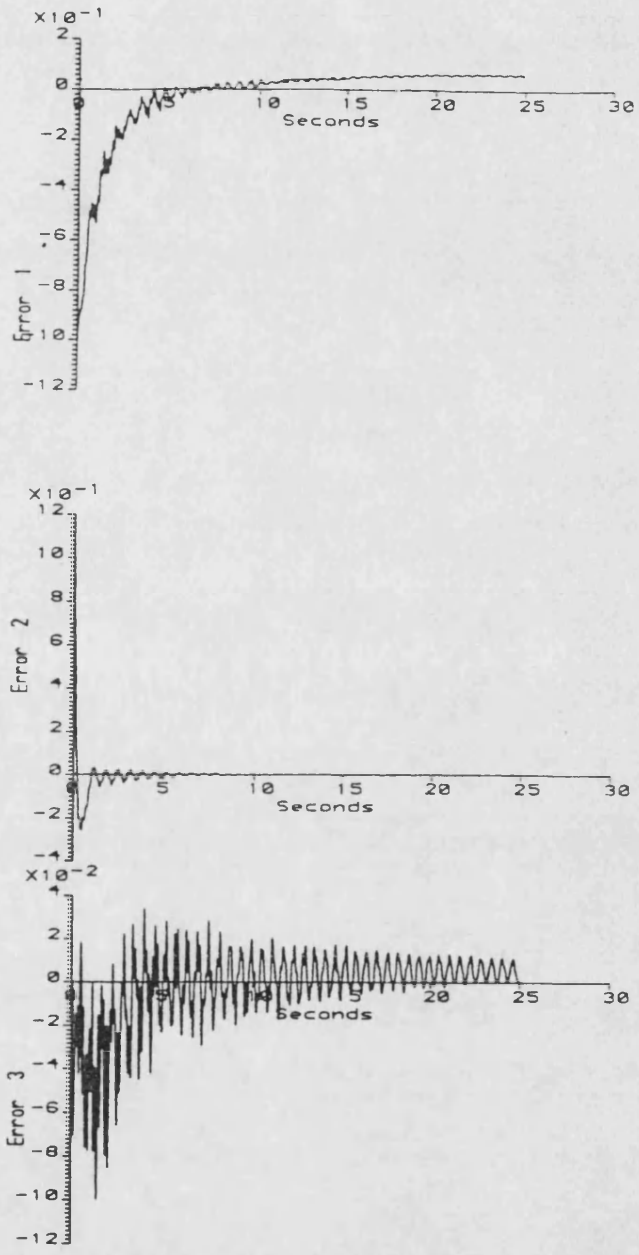


Figure 9.52: Error responses of input 2 of Platform attitude model with 26th order controller

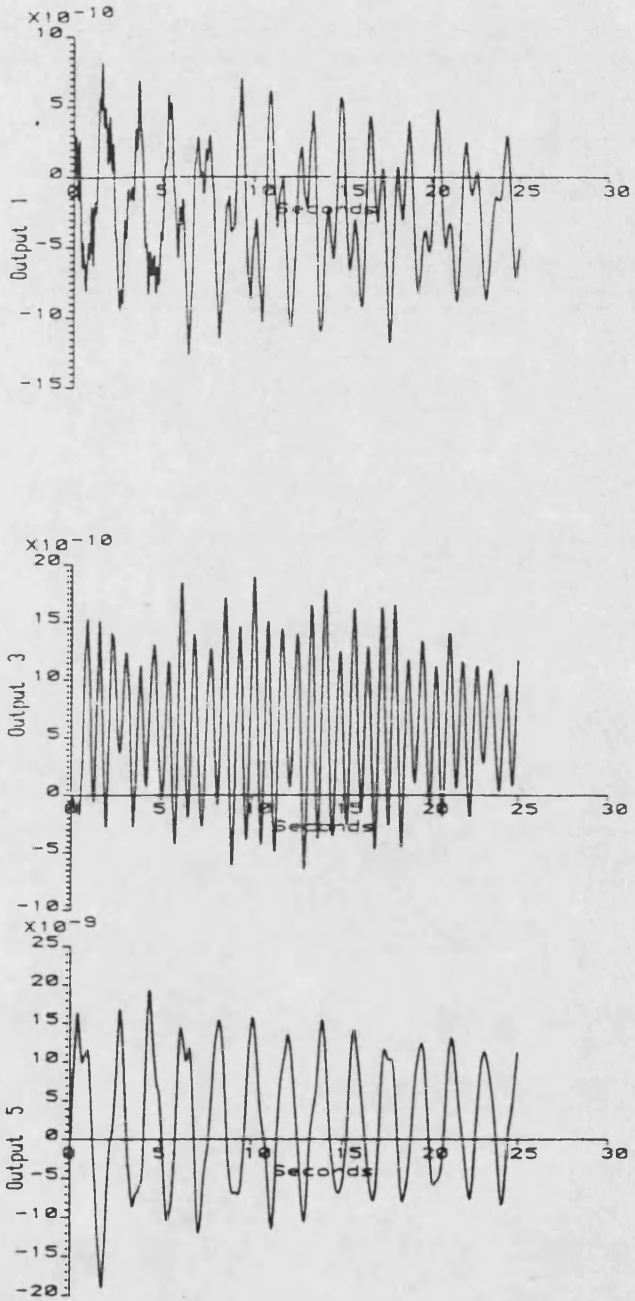


Figure 9.53: Attitude responses of input 3 of Platform attitude model with 26th order controller

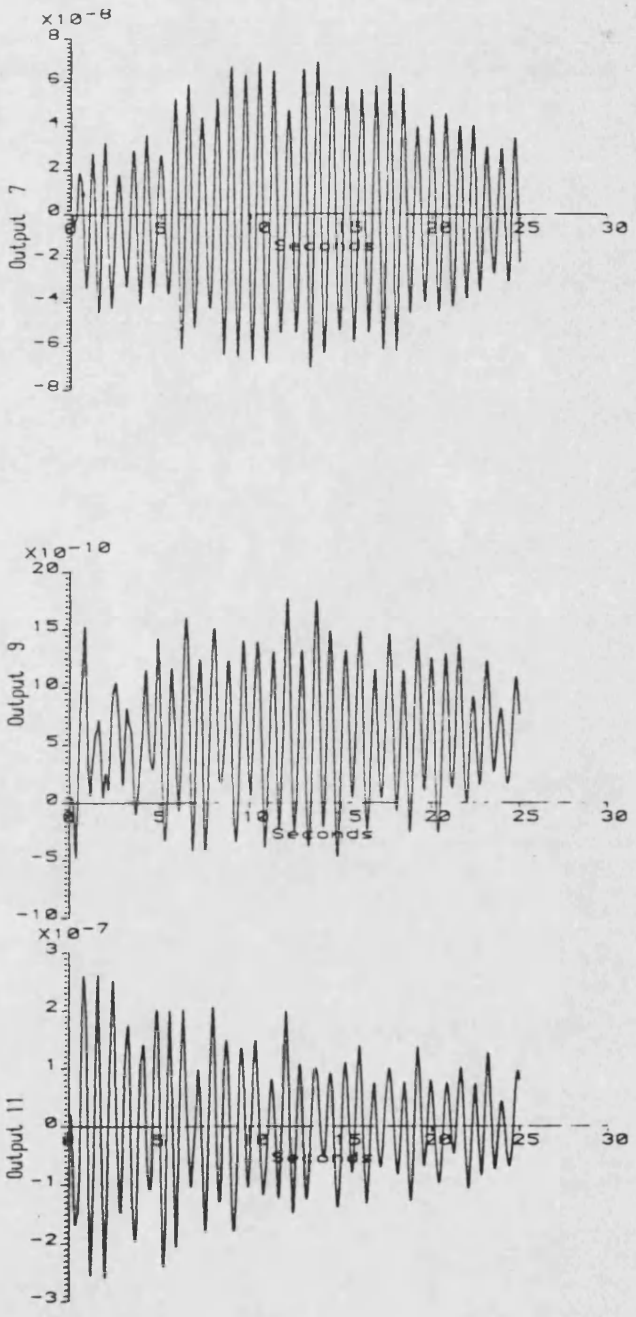


Figure 9.54: Payload responses of input 3 of Platform attitude model with 26th order controller

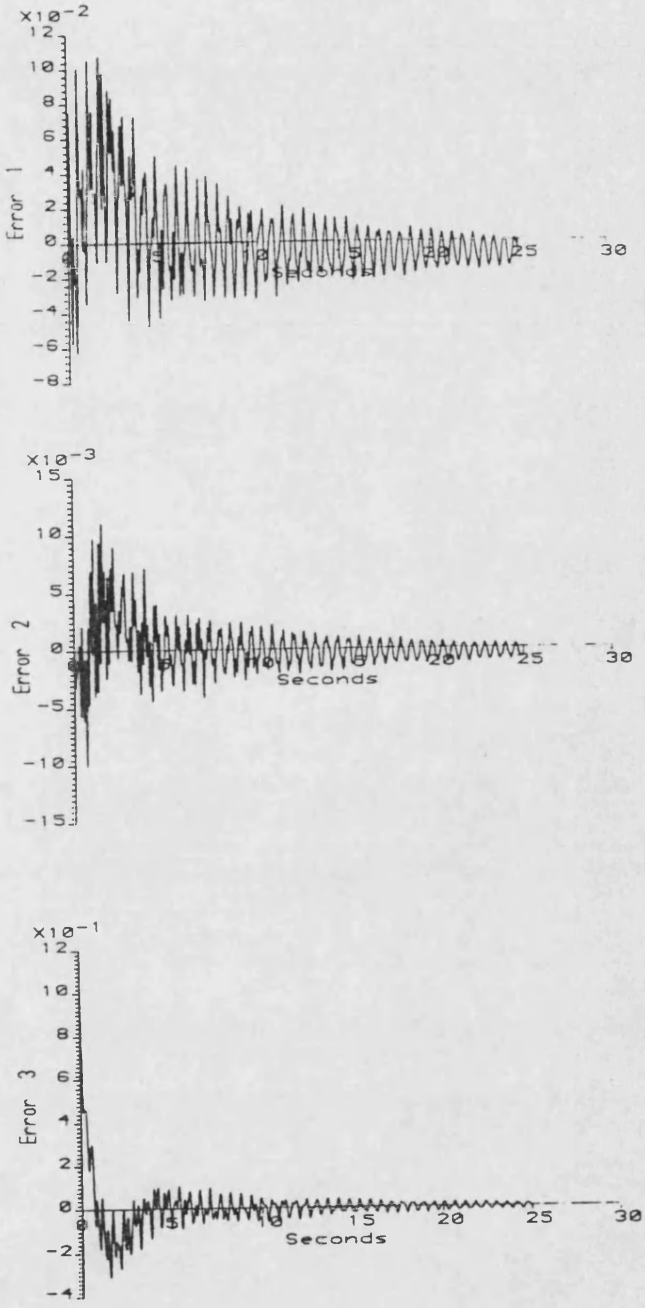


Figure 9.55: Error responses of input 3 of Platform attitude model with 26th order controller

Designed Eigenvalues Of Plant		Eigenvalues Of Closed Loop System	
Real	Imaginary	Real	Imaginary
-4.0656	4.0656	-0.11522E+07	0.00000E+00
-4.0656	-4.0656	-0.93407	61.215
-1.1435	1.1437	-0.93407	-61.215
-1.1435	-1.1437	-2.1975	54.367
-0.15625	0.15128	-2.1975	-54.367
-0.15625	-0.15128	-52.127	0.00000E+00
		-0.87836	38.317
		-0.87836	-38.317
		-0.71216	36.418
		-0.71216	-36.418
		-0.15103	29.482
		-0.15103	-29.482
		-0.34000	29.362
		-0.34000	-29.362
		-0.32596	24.453
		-0.32596	-24.453
		-0.22775	26.349
		-0.22775	-26.349
		-0.62632	19.457
		-0.62632	-19.457
		-11.954	9.7325
		-11.954	-9.7325
		-0.41403	12.603
		-0.41403	-12.603
		-0.58449E-01	10.114
		-0.58449E-01	-10.114
		-0.38054	9.5669
		-0.38054	-9.5669
		-0.56062E-01	9.2787
		-0.56062E-01	-9.2787
		-0.44717E-01	8.5031
		-0.44717E-01	-8.5031
		-0.84894E-01	8.8518
		-0.84894E-01	-8.8518
		-2.8130	5.1058
		-2.8130	-5.1058
		-0.31832	4.9988
		-0.31832	-4.9988
		-0.15298E-01	3.0625
		-0.15298E-01	-3.0625
		-1.2506	0.00000E+00
		-0.59693	0.00000E+00
		-0.16257	0.20035
		-0.16257	-0.20035
		-0.64749E-05	0.32249E-01
		-0.64749E-05	-0.32249E-01

Table 9.19: Poles of platform attitude model with output feedback controller

9.4.5 Dual Basis Controller

Several authors have proposed controllers based on a combination of state feedback and output feedback (see for example [115], [116]). The philosophy behind this proposal is that the desirable stability properties but generally lower performance capabilities of output feedback controllers could be combined with the generally more problematic but better performing state feedback based controllers. The output feedback is associated with a relatively low gain, wide bandwidth controller which acts to produce a small increase in the stability margin of a large number of modes, and the state feedback is associated with a relatively high gain, narrow bandwidth controller which acts to produce significant pole shifts but only on a small number of modes. The two sections of the controller are generally referred to as low authority control (the low gain output feedback), and high authority control (the high gain state feedback). The effect of the dual basis controller on the closed loop system is now examined analytically, initially for the ideal case.

9.4.5.1 Effect of Dual Basis Controller on Closed Loop System

Consider the following state space equation description of a linear system;-

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned}$$

subject to a control law of the form;-

$$u(t) = -F_s\hat{x}(t) - F_o y(t)$$

where F_s is the state feedback gain matrix (high authority control), and F_o is the output feedback matrix (low authority control). As usual, consider the states to estimated using a dynamic state estimator of the form:-

$$\dot{\hat{x}}(t) = A\hat{x}(t) + Bu(t) + H(y(t) - C\hat{x}(t))$$

The equations defining the overall closed loop system are:-

$$\begin{pmatrix} \dot{x}(t) \\ \dot{\hat{x}}(t) \end{pmatrix} = \begin{bmatrix} (A - BF_oC) & -BF_s \\ (HC - BF_oC) & (A - HC - BF_s) \end{bmatrix} \begin{pmatrix} x(t) \\ \hat{x}(t) \end{pmatrix}$$

Defining the state estimator error $e(t)$ as;-

$$e(t) = \hat{x}(t) - x(t)$$

the system equations can be written in the form;-

$$\begin{pmatrix} \dot{x}(t) \\ \dot{e}(t) \end{pmatrix} = \begin{bmatrix} (A - BF_oC - BF_s) & -BF_s \\ 0 & (A - HC) \end{bmatrix} \begin{pmatrix} x(t) \\ e(t) \end{pmatrix}$$

Examining the above equations, it can be seen that in this ideal case the addition of an output feedback term only affects the poles of the plant, having no effect on the poles of the state estimator. Thus, providing the conditions regarding the nature of the output feedback gain matrix as detailed in Chapter 4 are met (see also Appendix A.2), then the stability margin of the poles of the plant can be increased without affecting the poles of the state estimator.

This is the basis for the high authority/low authority controller. The low authority control (the output feedback term) is designed to provide a small increase in the stability margins of a large number of modes of the plant, whereas the high authority control (the state feedback term) is designed to provide more significant pole shifts for a small number of critical modes.

An obvious question arises as to which term should be designed first, the high authority control, or the low authority control. If the high authority control is designed first, it may be possible to select the low authority control to counteract any destabilising effects of "spillover" in unmodelled modes. However, assuming that this would be possible, it would almost certainly be necessary to redesign the state estimator to allow for the new positions of the plant poles. If the low authority control is designed first, then it is immediately obvious that the additional stability margin created may not be sufficient to cover the spillover problems created by the high authority control subsequently designed. Merely increasing the level of the low authority control is not generally suitable, as it has been shown (see simulation results shown earlier, and see also [117]) that this tends to improve the behaviour of the structure at actuator and sensor locations but has detrimental effects on other parts of the structure. This suggests that some form of iterative approach may be needed in order to achieve the necessary compromise.

Another question arises as to the behaviour of such a controller in non-ideal circumstances, such as exist for flexible spacecraft, and this is now examined in the same manner as used earlier.

Consider the system defined by the following equations;-

$$\begin{pmatrix} \dot{x}_r(t) \\ \dot{x}_n(t) \end{pmatrix} = \begin{bmatrix} A_r & 0 \\ 0 & A_n \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \end{pmatrix} + \begin{bmatrix} B_r \\ B_n \end{bmatrix} u(t) \quad (a)$$

$$y(t) = \begin{bmatrix} C_r & C_n \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \end{pmatrix} \quad (b)$$

which is subject to a control law of the form;-

$$u(t) = -F_o y - F_s \hat{x}_r(t)$$

where $\hat{x}_r(t)$ is an estimate of the state $x_r(t)$ obtained via a state estimator of the form;-

$$\dot{\hat{x}}_r(t) = A_r \hat{x}_r(t) + B_r u(t) + H\left(y(t) - C_r \hat{x}_r(t)\right)$$

The equations defining the closed loop system are given by;-

$$\begin{pmatrix} \dot{x}_r(t) \\ \dot{x}_n(t) \\ \dot{\hat{x}}(t) \end{pmatrix} = \begin{bmatrix} (A_r - B_r F_o C_r) & -B_r F_o C_n & -B_r F_s \\ -B_n F_o C_r & (A_n - B_n F_o C_n) & -B_n F_s \\ (HC_r - B_r F_o C_r) & (HC_n - B_r F_o C_n) & (A_r - HC_r - B_r F_s) \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \\ \hat{x}_r(t) \end{pmatrix}$$

or alternatively, using the usual definition for the error function $e(t)$, as follows;-

$$e(t) = \hat{x}_r(t) - x_r(t)$$

the equations can be written in the form;-

$$\begin{pmatrix} \dot{x}_r(t) \\ \dot{x}_n(t) \\ \dot{e}(t) \end{pmatrix} = \begin{bmatrix} (A_r - B_r F_s - B_r F_o C_r) & -B_r F_o C_n & -B_r F_s \\ (-B_n F_o C_r - B_n F) & (A_n - B_n F_o C_n) & -B_n F_s \\ 0 & HC_n & (A_r - HC_r) \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \\ e(t) \end{pmatrix}$$

It can be seen from the above equations that the interaction between the two control terms is more complex than in the ideal case, and in particular, the independence of the poles of the state estimator can no longer be assumed. As the effect of such a dual basis controller on the closed loop poles of the system is not immediately obvious, an exercise was carried out to attempt to learn something of this effect, and this is now discussed.

9.4.5.2 Design Exercise

In this exercise only rate feedback was added to the system via a scaled unity feedback gain matrix. The program HACLAC (see Chapter 7) was used to compute state feedback and state estimator gain matrices for a reduced order design model, and then compute the closed loop poles of the system formed by the controller and the full order model. Subsequently, a rate feedback term was added to the control law, and the poles of the closed loop system were recomputed. The scaling term for the output feedback matrix was then increased, and the poles of the closed loop system recomputed. The scaling term was further increased by an order of magnitude each time, and the poles recomputed at each stage. The scaling term was varied over six orders of magnitude.

This procedure was carried out for designs based on reduced order models of 10th order, 14th order, 18th order, 22nd order, and 26th order. The numerical details of the results would occupy a volume in their own right, so are not included here, but the trends identified from them are discussed.

The results of the program HACLAC showed that whilst the addition of rate output feedback subsequent to state feedback based controller design did increase the stability margin of some of the closed loop poles, it had the opposite effect on others. Hence what was gained on one hand was lost on the other.

Consequently, the design process was revised so that the output feedback term was added to the plant first, and then the state feedback based controller was designed with a reduced order model which takes into account the output feedback term. This was done using the program LACHAC (see Chapter 7) in order to see if the effects of the revised procedure were more beneficial. The same series of reduced order models and range of rate feedback gains were used, and again the numerical details of the results are not included due to their volume. However, the underlying trends are the same as for the previous case, that is where some of the poles have increased stability margins as expected, others have decreased stability margins.

These results have suggested that there is no real benefit to be gained from using such dual basis controllers for flexible spacecraft, particularly when a reduced order model is used in the design of the state feedback based controller.

9.4.6 Remarks

The attitude controller design exercise has provided much greater insight into the difficulties associated with the design of "high performance" controllers for large flexible spacecraft where the required controller bandwidth is large allowing elastic modes to intrude.

Again, the design of state feedback based controllers using "optimal" design techniques (LQG) has been shown to be both practical and successful. The use of reduced order design models obtained using modal cost analysis as a basis for these procedures has also been shown to be practical. However, the choice of a suitable order for the reduced order design model is not definable *a priori*, and consequently an iterative approach to controller design has to be adopted. In addition, the design procedure itself requires an iterative approach in order to refine the choice of weighting terms. As the computational requirements of these procedures exceed the limit of hand calculation, even for apparently trivial examples, the use of a digital computer to carry out the computations is essential, and for practical problems such as the one considered in this chapter the computing requirements are extremely large, and consequently expensive.

Another aspect of state feedback based controllers that has become more apparent in this chapter is the effect of using a reduced order design model on steady state errors. Recall from Chapter 4 the equations defining the closed loop system incorporating state feedback and a dynamic state estimator in the ideal case as;-

$$\begin{pmatrix} \dot{x}(t) \\ \dot{\hat{x}}(t) \end{pmatrix} = \begin{bmatrix} A & -BF \\ HC & (A - HC - BF) \end{bmatrix} \begin{pmatrix} x(t) \\ \hat{x}(t) \end{pmatrix}$$

When the system is in steady state, that is $\dot{x}(t) = \dot{\hat{x}}(t) = 0$, these equations become;-

$$\begin{aligned} Ax_{ss} &= BF\hat{x}_{ss} \\ HCx_{ss} &= -A\hat{x}_{ss} + BF\hat{x}_{ss} + HC\hat{x}_{ss} \end{aligned}$$

and therefore;-

$$\hat{x}_{ss} = x_{ss}$$

Consequently the steady state error in the estimates of the states is zero. However, now recall from Chapter 4 that the equations defining the closed loop system incorporating state feedback and a dynamic state estimator

where a reduced order model has been used to design the state estimator are as follows;-

$$\begin{pmatrix} \dot{x}_r(t) \\ \dot{x}_n(t) \\ \dot{\hat{x}}_r(t) \end{pmatrix} = \begin{bmatrix} A_r & 0 & -B_rF \\ 0 & A_n & -B_nF \\ HC_r & HC_n & (A_r - HC_r - B_rF) \end{bmatrix} \begin{pmatrix} x_r(t) \\ x_n(t) \\ \hat{x}_r(t) \end{pmatrix}$$

When this system is in steady state, that is $\dot{x}_r(t) = \dot{x}_n(t) = \dot{\hat{x}}_r(t) = 0$, these equations become;-

$$\begin{aligned} A_r x_{r_{ss}} &= B_r F \hat{x}_{r_{ss}} \\ A_n x_{n_{ss}} &= B_n F \hat{x}_{r_{ss}} \\ 0 &= HC_r x_{r_{ss}} + HC_n x_{n_{ss}} + (A_r - HC_r - B_rF)\hat{x}_{r_{ss}} \end{aligned}$$

and therefore;-

$$-HC_r x_{r_{ss}} = \left(HC_n A_n^{-1} B_n F + A_r - HC_r - B_rF\right) \hat{x}_{r_{ss}}$$

Hence if;-

$$HC_n A_n^{-1} B_n F \neq 0$$

then;-

$$\hat{x}_{r_{ss}} \neq x_{r_{ss}}$$

Note that for practical cases the matrix A_n^{-1} will exist because A_n corresponds to the neglected modes and thus none of the poles would be zero because poles at the origin of the complex plane correspond to rigid modes and these must not be neglected from a design model (see Chapter 3). This analysis shows how the estimates of the states of the design model will contain steady state errors and the effects of these errors can be observed in the simulation results presented earlier. It has been identified that the term which gives rise to these errors is $HC_nA_n^{-1}B_nF$ and some aspects of this term are now examined.

Note that the matrix H is the state estimator gain matrix and the matrix F is the state feedback gain matrix, and that if the required controller performance is high this will lead to large values in both H and F, tending to compound the steady state error problem. Note also that the central part of the term, that is $C_nA_n^{-1}B_n$, is in fact the steady state transfer function matrix of the neglected part of the system, that is the steady state gain matrix of the neglected subsystem. Thus if this gain is low compared to the steady state gain of the retained subsystem, then the steady state estimator

error will also be low. Continuing further, the steady state gain of the neglected subsystem will be low compared to the steady state gain of the retained subsystem if the model error associated with the retained subsystem (the design model) is low (see earlier sections and Chapter 3). Thus the successful design of a state feedback based controller for a flexible spacecraft, both in terms of minimising steady state errors in the state estimator and in minimising spillover problems, requires a model error associated with the design model which is as close to zero as possible.

The design of an output feedback based controller for the platform attitude model demonstrated how the application of high gain output feedback to flexible spacecraft also has its problems. The extremely large pole shifts associated with this controller would of course be moderated when the finite bandwidth of practical actuators and sensors are considered, but the phase shifts associated with non-ideal devices could also lead to stability problems. It is highly likely that better results could be obtained if some allowance was made for the effects of the elastic modes which were completely ignored during the design procedure.

9.5 Summary

The use of the simple beam in the previous chapter as a basis for the evaluation of control system design methods, although useful, was not truely representative of a practical problem, and thus a more realistic example has been used in this chapter to give a better insight into the practical difficulties associated with a complex structure. Thus an early version of the ESA Space Platform was introduced as a realistic example of a large flexible spacecraft and the derivation of a mathematical model was described.

Subsequently, a baseline controller was designed for full six degree of freedom control which is based on full order state feedback and an appropriate state estimator. This design was used as a reference for comparing the performance of subsequent reduced order controllers, based on state feedback or output feedback.

The remarks following this design exercise indicate its shortcomings and outline the aims of the second design exercise, which considers the design of a "high performance" attitude controller for the Space Platform. Again, a baseline controller is designed to act as a reference for comparision, and subsequently several methods are used to design reduced order controllers, although less successfully than in the previous design exercise.

An aspect of the use of reduced order models in the design of state feedback based controllers has also become more apparent in this exercise, and that is the problem of steady state errors. The simple analysis shows that to minimise these errors the model used for design purposes must be as accurate as possible, which generally implies the use of a relatively high order model.

Also considered in this exercise are what has been termed "dual basis" controllers, that is controllers based on both output feedback and state feedback. Although in the "ideal" case, this type of controller is shown to appear to offer the advantages of both techniques, in the practical case the disadvantages of both techniques dominate the resulting controller, and consequently this type of controller seems to be of little use for flexible spacecraft.

Chapter 10

Conclusions

10.1 Modelling

Three methods of representing the dynamics of flexible structures have been considered in Chapter 2, referred to as the hybrid co-ordinate approach, the multibody approach, and the finite element approach. The hybrid co-ordinate approach and the finite element approach are both limited to small elastic deformations, whereas the multibody approach is not, but is generally more difficult to define data for the representation of a real structure.

The hybrid co-ordinate approach is particularly useful for dealing with the attitude control problem of a spacecraft when the flexing of the structure between the actuators and sensors is negligible. The finite element approach can also be used for modelling the attitude control problem, and has the added advantage of being able to cope with the problem of structural flexing between actuators and sensors, and it can also be used to model other parts of the structure which are not associated with the point of attitude control (see Chapter 9), such as communications antennae or payloads for example.

If the numerical data defining the elastic modes has been obtained utilising any artificial constraints, for example modal data representing the characteristics of a solar array where the root of the array has been assumed to be constrained so that no motion can take place, then extreme caution should be exercised over its use. If the structure to which the above mentioned array is mounted happens to have much greater inertia than the array, then the artificial constraint is not too severe an approximation, and the resonant frequencies of the overall structure will not differ greatly from those of the array. However, if the main structure has inertia of the same or-

der, or lower, than that of the array, then the artificial constraint is seriously violated. In this case, significant differences between the actual frequencies of the structure and those predicted from the array data can occur, leading to large modelling inaccuracies. Thus if it is required to obtain modal data for a flexible structure which is based on a set of flexible substructures, then it is far more accurate to construct the model from "physical" data (for instance, a set of finite element representations) and then to obtain modal data for the complete structure, rather than attempt to convert the modal data for each substructure.

Another advantage of the finite element method is the fact that most finite element structural analysis packages incorporate the facility to compute stresses in the various elements. This information would allow some optimisation of the structural components, possibly leading to weight reduction, an important aspect of spacecraft design.

Careful consideration of fundamental mechanics at an early stage in the proposal of configurations for large spacecraft could also avoid many problems relating to structural flexing. For instance, consider the Space Platform described in Chapter 9. The modal analysis of the structure revealed an unexpectedly low resonant frequency of the structure at about 1.28Hz which was associated with the main structure not the known flexible substructures such as the solar arrays, and analysis of the modeshape data showed that this mode corresponded to torsional motion of the central 4.5m main beam, together with a pendulum motion of the 3rd payload and the 3.0m beam. Obviously, this resonant frequency could be made much higher either by stiffening the 4.5m tube, which would incur a weight penalty, or by reducing the length of the 3.0m tube. (Reducing the payload mass is another option, but unlikely to be favourable to customers.) The most useful solution in this case is to make the spacecraft more compact, and this approach has been adopted in later configuration proposals.

10.2 Controller Design

10.2.1 Approaches

The reason for considering the design of reduced order controllers is two-fold. Firstly, the infinite dimensional nature of the dynamics of a flexible structure means that any finite dimension model used to represent the structure must inherently be a reduced order approximation. Secondly, the aim of control systems design is to produce a controller which provides the required per-

formance from the system but which is as simple as possible. With state feedback based design techniques the resulting controller is of the same (or similar) order as that of the design model. This may cause implementation difficulties due to the generally high order of models of flexible structures, thus the derivation of controllers of lower order which produce similar performance is obviously of great interest.

Ideologically, the derivation of an "equivalent" lower order controller from a high order controller is highly desirable, but none of the methods examined here produced satisfactory results. The technique known as the "optimally projected equations" approach (see Chapter 3) was not examined in detail due to the problems associated with the solution of the various equations, but this is an obvious area for further study.

The model reduction method of modal cost analysis has been found to be easy to use, and to produce useful results. The facility to compute a "figure of merit" (the model error term) for the subsequently derived reduced order model is most useful. Although this figure cannot be used in an absolute manner, it does give some indication as to the susceptability of the system to the effects of spillover (see Chapter 4) caused by the use of the reduced order design model, and certain points regarding the use of this method are now given.

- For a given plant, if the model error term is small, then the effects of spillover tend to be small, but the controller order is high.
- For a given plant, if the required system response is fast, this implies high gains, so a small model error term is essential for successful design, and even more so for small steady state errors in the system.

Obviously for a given plant, a compromise has to be sought between controller complexity, speed of closed loop system response, the effects of spillover, and the effects of steady state errors. This has been demonstrated in Chapter 9, where this technique was applied to a model of a large flexible spacecraft.

10.2.2 Design Methods

Of the various design methods discussed in Chapter 5, by far the majority are not particularly useful for designing controllers for large flexible spacecraft. The primary reason for this is the numerical difficulties experienced by many of the methods when applied to problems greater than about 10th order.

This figure is of course associated with the type of models studied here, and so may not be representative of all systems.

The method of independent modal space control is only useful for a particular (and probably very rare) case where the number of independent actuators equals the number of independent sensors which equals the number of modes to be controlled. Note that this method does not give any guide as to the number of modes that will need to be controlled, and if the number of modes chosen is too small the model will be a poor representation of the actual plant and the system will suffer from the effects of spillover. The use of pseudo-inverse matrices as a means of "observing" the system states was found to be unsatisfactory for moderately "non-square" examples (that is where the number of states exceeds the number of outputs), and totally impractical for very non-square problems.

The algorithms used for obtaining the controllable canonical form and the observable canonical form of state space models were found to be unsuitable for high order problems (above about 10th order for problems of the type considered here), or numerically stiff problems, thus their range of application is rather limited.

The significance of spillover effects due to the use of a reduced order model for design purposes is currently not predictable *a priori* for a particular case. Currently the only means of evaluating spillover effects is to compute the poles of the closed loop system using the most accurate model available (which usually means the highest order). Some method of predicting the possible effects of spillover caused by the use of a particular reduced order model would be highly desirable.

The most useful method of designing state feedback based controllers (the design of both the state feedback gain matrix and the state estimator gain matrix) is that based on the steady state solution of the linear quadratic gaussian (LQG) problem. The various weighting terms can be used to influence the characteristics of the resulting controller (see Chapter 5), and the algorithm used gave reliable results even for high order problems. The most complex example attempted, although not detailed in this work, was 90th order, and the design was successful.

The use of this approach for the design of dynamic state estimators has been found to be most successful. A particular problem with other state estimator design techniques is that they generally rely on a pole placement algorithm (the associated numerical difficulties of which have already been referred to) which inherently require *a priori* knowledge of the desired pole locations. Whilst certain guidelines can be given as to the general location

of such poles for reasonable performance in the ideal case (see Chapter 5), guidance for high order problems is sparse, and the effects of pole locations on aspects such as robustness (that is performance in the non-ideal case) needs further investigation.

The performance achievable by the use of an output feedback based controller appears to be limited, but the primary advantage of these controllers is the relatively simple implementation.

Dual basis controllers (as discussed in Chapter 9) appear initially to promise the best of both worlds, but in practice they are plagued by the drawbacks of both methods, and thus offer little (if any) advantage over a more conventional scheme.

It should always be remembered that the aim of the controller design is to produce a controller that gives the required performance but which is as simple as possible. Thus, even though a state feedback based design with its attendant state estimator may be considered by some to be a more elegant solution, if suitable performance can be achieved from an output feedback based controller then the advantages of simpler implementation of the output feedback design should not be forgotten.

10.2.3 Effects of Actuators and Sensors

Almost exclusively throughout this work the actuators and sensors have been considered to be ideal, that is of infinite bandwidth and completely linear with a gain of unity. The main reason for this is due to the range of available devices with vastly differing characteristics, and the fact that the choice of actuators and sensors for a particular spacecraft depend to some extent on the mission requirements of the vehicle (see for example [5]).

Another aspect of actuators that has to be considered in the design of controllers for large spacecraft is the level of control energy that they can supply to the spacecraft. Large spacecraft generally have large moments of inertia, and if fast responses are sought then powerful actuators will be necessary.

The essential elements of a high performance control scheme will be high performance actuators and sensors, and a high performance computing system, the latter most likely to be digital. Good sensors such as gyroscope units are currently available with suitable response times (although they generally require additional sensors to correct long term drift effects), and high performance high reliability digital computing systems have recently been demonstrated [118] which could be used for the implementation of

controllers. The limiting factor on the overall system performance appears likely to be the actuator devices, as the ability to faithfully convert the demands of the controller into torques or forces on the spacecraft is essential. Currently, the only nominally linear devices with reasonably high torque capabilities are control moment gyros (CMG's), or reaction wheels, but the suitability of these devices to coping with very large flexible spacecraft would require further investigation.

Another aspect that has not been considered here is the effects of sensor noise. High performance/wide bandwidth control systems tend to be rather sensitive to noise, and thus the sensor noise terms may be a limiting factor on the pointing performance achievable with particular hardware.

10.3 Summary

This work has shown that it is possible to design controllers for large flexible spacecraft although the choice of methods is rather limited for complex high order problems. If the required performance can be obtained with a controller cut-off frequency well below the frequency of the elastic modes then the design of the controller is relatively simple and the presence of the elastic modes do not cause much concern.

However, if the required performance can only be achieved with a controller bandwidth which extends into the frequencies of the elastic modes, then the design is less straightforward, requiring much more exact consideration of the elastic modes, particularly those modes within or close to the controller bandwidth. The design of high performance state feedback based controllers is best carried out using linear quadratic gaussian (LQG) techniques, and modal cost analysis is a practical method for deriving reduced order models for design purposes, although the demise of the Separation Theorem caused by the use of reduced order models, and the effects on steady state errors must be borne in mind.

A technique used frequently in industry is to consider the three axes of the spacecraft to be completely decoupled and to design three separate single axis controllers. It should be noted that one of the effects of flexure dynamics is to increase the interaction between axes, both statically and dynamically, and this can give rise to stability problems for controllers designed in this manner. However, it has been shown [119] that if proportional-integral-derivative (PID) controllers designed in this manner using rigid mode dynamics alone are unconditionally stable with respect to gain, then they are

also unconditionally stable with respect to flexure dynamics. Also, past spacecraft have generally been designed such that the principal axes of inertia are nominally aligned with the flight axes, thus assisting the decoupled axes approach. As future designs become more complex, the ability to balance the spacecraft in such a manner may be reduced, and thus more account of axis interaction will become necessary. If such difficulties arise, then it is suggested that some of the methods examined here may be utilised to tackle the problem.

Chapter 11

Further Work

11.1 Modelling

To be able to avoid the small displacement limitations of the hybrid coordinate approach and the finite element approach, it would be useful to investigate the possibility of combining one of these approaches with the multibody approach. Besides being better able to cope with large displacements as well as small displacements, it would be convenient for practical spacecraft analysis because the various substructures that form a spacecraft (such as solar arrays, large antennae, payloads, *etc.*) which are often designed in comparative isolation, could be easily and quickly brought together for accurate analysis of the complete structure.

11.2 Controller Design

11.2.1 Techniques

An obvious area for further study is to examine the various frequency domain techniques for model order reduction and control system design. It has been noted in Chapter 5 that a practical investigation would require an interactive computer-aided design facility with not inconsiderable graphics capabilities. Techniques based on inverse Nyquist arrays [86], [87], or characteristic loci [88], [89], are obvious candidates for further investigation.

It would also be interesting to examine further the linear quadratic qaussian technique, in particular to consider frequency weighted cost functions as proposed in [115]. Also related to the LQG technique, it would be desirable to study the effects of pole locations on the performance of dynamic state

estimators, considering such aspects as sensitivity to parameter variations, pole groupings, and noise in the system.

The effect of reduced order design models on steady state errors in dynamic state estimators could also be examined further. Preliminary studies (not detailed here) suggest that these errors could largely be erradicated by the inclusion of an additional d.c. (steady state) term in the reduced order model which allows for the effect of the neglected modes. It is possible that the work by Scott [120] may be extended to provide a simple means of calculating the correction term.

Other techniques that have not been examined here but might prove to be useful are variable structure control (see for example [121]), and various methods of adaptive control (see for example [63], [122], [123], [124], [125]). If the theory of variable structure control could be extended to cover the case of switching with a deadband, then this would be an ideal method to use for tackling the highly non-linear problem of designing spacecraft control schemes based on thrusters, which are essentially on/off devices. Such schemes are usually designed to operate in a single-sided limit cycle mode where the spacecraft gently oscillates between certain attitude error limits, which are defined by the attitude pointing requirements of the mission. The ability to accurately define the control law required to produce a particular limit cycle would be extremely useful.

Adaptive control schemes might prove useful in coping with the problems associated with model errors and parameter variations. Significant model errors can arise simply due to the difficulty of defining the properties of certain structures, such as the blankets on which solar cells which form the solar arrays are mounted. Thus control systems generally have to be rather conservative in order to be reasonably insensitive to the model used for design purposes. An example where parameter variations can occur is also linked to solar arrays. Generally, the solar arrays on a spacecraft have to be maintained normal to the sun vector to obtain maximum efficiency of the solar cells, and this implies for many spacecraft (but not all, as it depends on the orbit of the spacecraft) that the solar arrays have to rotate relative to the rest of the spacecraft and this has the effect of causing a cyclic variation in the resonant frequencies and modeshapes of the spacecraft. This variation has a period equal to the orbit period (for most Earth pointing spacecraft) and so is generally much slower than the attitude dynamics and consequently is usually treated as a range of modelling errors. The comments above thus also apply to this case, and hence the controller design is usually made rather conservative. It is possible that better closed loop performance

may be attainable using some form of adaptive controller (perhaps based on an extended Kalman filter [126], or an adaptive observer [18], [127]) which updated the control laws "on-line" with knowledge of current parameter values.

11.2.2 Stability and Robustness

Some better form of stability analysis would be most useful, particularly with high order time domain designs, together with a robustness analysis method. The method of conic sectors used by Safonov [113] might prove fruitful, or possibly the current interest in singular values for stability and robustness measures (see for example [128],[129]) might produce some useful results.

11.2.3 Numerical Methods

The problems of obtaining the controllable canonical form or the observable canonical form of high order or numerically stiff state space models with the algorithm used here have been pointed out many times. If an alternative algorithm could be found which could overcome these difficulties, it would allow the application of pole placement methods to such problems.

Another numerical problem highlighted but not tackled in this work is the solution of the "optimally projected equations" problem (see Chapter 5). The solution of this set of equations is not a trivial problem, but if a reliable method could be found (a method has been proposed [52], but this has not been examined) the results it produces may prove to be most useful as another means of designing reduced order state feedback based controllers.

Methods of designing output feedback based controllers using parameter optimisation techniques have been proposed by several authors (see for example [130],[45]), however the algorithms required to solve such problems are generally complex and require careful design to produce reliable results, but they might prove to be more suitable for high order problems than the methods considered here.

The use of the Numerical Algorithms Group (NAG) mathematical routines (see Appendix B) has proved to be of great assistance in the development of reliable control systems design programmes, and an extension to the NAG library has been proposed [131] which will contain many useful routines for control systems design. This extension of the NAG library should greatly ease the problems of implementing reliable design programmes.

11.2.4 Actuators and Sensors

The effects of finite bandwidths and non-linearities of actuators and sensors needs to be considered. Although a little work has been done to examine these aspects [68], because of the vastly differing characteristics of the various devices available an abstract analysis may be of limited value, and consequently it may be that this aspect can only be properly addressed when a specific spacecraft is examined where the various devices will be stipulated.

Another aspect of interest concerning actuators is that they are generally considered as point source devices. The effect of non-point source devices has been considered by Hablani [132], but this work has not been examined in detail.

The effects of noise on the performance of wide bandwidth attitude control systems needs further investigation, because any practical control devices will produce noise and the level of this noise may be critical for high precision attitude control systems.

The question of where on a structure to locate the actuators and sensors has received some attention (see for example [81], [133]), but this is probably not very relevant to the problem of attitude control of spacecraft. The reason for this is the fact that attitude control is essentially about the pointing of one particular part of the spacecraft, and common sense dictates that the actuators and sensors should be mounted at, or as close as possible to, that point of the spacecraft from which the attitude errors are defined. Also, the fact that most spacecraft tend to some extent to be modular in design means that the location of the actuators and sensors is frequently defined well before the control law is designed, so little, if any, freedom is available to the analyst.

However, "shape" control problems, for instance the maintenance of the shape of a large lightweight antenna reflector, may be less restrictive allowing some choice of device locations, in which case the facility to define the device locations may be useful to attempt to minimise the energy expended by the controller to achieve the required control.

11.3 Real–Time Control Demonstration

The completion of the experimental rig would enable demonstration of the implementation of proposed controllers thus providing convincing evidence of the practicality of such schemes. Although the restrictions of such Earth-based experiments limit the ability to examine completely realistic exam-

ples of flexible spacecraft, this would demonstrate the ability of proposed controllers to cope with implementation effects such as quantisation, non-linearities, and noise.

An alternative approach which would offer more scope for considering realistic examples of flexible spacecraft would be to apply the implemented controller to the control of a sophisticated real-time simulation of a flexible spacecraft, using a "hardware in the loop" approach [134]. This is in fact the method usually adopted for the testing of flight hardware.

Appendix A

Mathematical Results

A.1 Derivation of the Equations of Motion of a Flexible Structure Using Finite Elements

Although the analysis of beams and plates by analytical methods is well documented (for example, see [93]), the analytical analysis of anything other than simple structures rapidly becomes very complex and thus error prone, as well as tedious. An alternative approach that can be used to obtain equations describing the motions of complex structures and that can be easily mechanised on digital computers is that of finite elements. With this technique, the structure is described by a finite set of elements, each element being described at its boundaries, with assumed shape functions describing the relative displacements within the boundaries. Thus the structure is discretised and the boundaries of the elements are known as the "nodes" of the structure. The finite element equations can be arranged to enable the motion of the structure at these nodes to be described. The derivation of the equations of motion of a structure using this representation is now described. (See also [30] and [31].)

Assume a displacement model of element e as;-

$$\overline{U}(x,y,z,t) = \begin{bmatrix} u(x,y,z,t) \\ v(x,y,z,t) \\ w(x,y,z,t) \end{bmatrix} = [N(x,y,z)]\overline{d}_e(t) \qquad (A.1)$$

where $\overline{U}$ is the vector of element displacements, $[N]$ is the matrix of shape functions, and $\overline{d}_e$ is the vector of nodal displacements. From equation A.1,

the strains in an element can be expressed as;-

$$\overline{E}(t) = [B]\,\overline{d}_e(t) \tag{A.2}$$

and the stresses as;-

$$\overline{\sigma}(t) = [D]\,\overline{E}(t) = [D]\,[B]\,\overline{d}_e(t) \tag{A.3}$$

By differentiating equation A.1 with respect to time, the velocity field can be obtained as;-

$$\dot{\overline{U}}(x, y, z, t) = [N(x, y, z)]\,\dot{\overline{d}}_e(t)$$

where $\dot{\overline{d}}_e$ is the vector of nodal velocities.

To derive the equations of motion, Lagrange's equations can be employed, which are given by;-

$$\frac{d}{dt}\left(\frac{\delta L}{\delta \dot{d}}\right) - \frac{\delta L}{\delta d} + \frac{\delta R}{\delta \dot{d}} = 0 \tag{A.4}$$

where $L = T - P$ is called the Lagrangian function, T is the kinetic energy function, P is the potential energy function, and R is the dissipation function. The kinetic energy of an element e can be expressed as;-

$$T_e = \frac{1}{2}\iiint_{V_e} \rho\, \dot{\overline{U}}^T \dot{\overline{U}}^T dV \tag{A.5}$$

and the potential energy as;-

$$P_e = \frac{1}{2}\iiint_{V_e} \overline{E}^T \overline{\sigma} dV - \iint_{S_e} \overline{U}^T \overline{\theta} dS - \iiint_{V_e} \overline{U}^T \overline{\theta} dV \tag{A.6}$$

where V_e indicates element volume, S_e indicates element surface, ρ is the density of the material of the element, $\overline{\theta}$ is the vector of prescribed forces. By assuming the existence of dissipative forces proportional to the relative velocities, the dissipative function of the element e can be expressed as;-

$$R_e = \frac{1}{2}\iiint_{V_e} \mu\, \dot{\overline{U}}^T \dot{\overline{U}} dV \tag{A.7}$$

where μ is known as the damping coefficient. In equations A.5, A.6, and A.7, the volume integral is to be taken over the volume of the element, and the surface integral is to be taken over that portion of the surface of the element

on which the surface forces are prescribed. Using equations A.1, A.2, and A.3, the expressions for T, P, and R can be written as;-

$$T = \sum_{e=1}^{E} T_e = \frac{1}{2}\dot{\overline{d}}_g^{\,T}\left[\sum_{e=1}^{E}\iiint_{V_e}\rho[N]^T[N]\,dV\right]\dot{\overline{d}}_g \tag{A.8}$$

$$\begin{aligned} P = \sum_{e=1}^{E} P_e &= \frac{1}{2}\overline{d}_g^{T}\left[\sum_{e=1}^{E}\iiint_{V_e}[B]^T[D][B]\,dV\right]\overline{d}_g \\ &\quad - \overline{d}_g^{T}\left[\sum_{e=1}^{E}\left(\iint_{S_e}\overline{N}^T\overline{\theta}\,dS + \iiint_{V_e}\overline{N}^T\overline{\theta}\,dV\right)\right] \\ &\quad - \overline{d}_g^{T}\overline{f}_c(t) \end{aligned} \tag{A.9}$$

$$R = \sum_{e=1}^{E} R_e = \frac{1}{2}\dot{\overline{d}}_g^{\,T}\left[\sum_{e=1}^{E}\iiint_{V_e}\mu[N]^T[N]\,dV\right]\dot{\overline{d}}_g \tag{A.10}$$

where $\overline{d}_g$ is the global node displacement vector, $\dot{\overline{d}}_g$ is the global node velocity vector, and $\overline{f}_c$ is the vector of concentrated forces at the nodes of the structure.

The following matrices and vectors can now be defined;-

$[M_e]$ = element mass matrix

$$= \iiint_{V_e}\rho[N]^T[N]\,dV$$

$[K_e]$ = element stiffness matrix

$$= \iiint_{V_e}[B]^T[D][B]\,dV$$

$[C_e]$ = element damping matrix

$$= \iiint_{V_e}\mu[N]^T[N]\,dV$$

$\overline{f}_{se}$ = vector of element nodal forces produced by surface forces

$$= \iint_{S_e}[N]^T\overline{\theta}\,dS$$

$\overline{f}_{be}$ = vector of element nodal forces produced by body forces

$$= \iiint_{V_e}[N]^T\overline{\theta}\,dV$$

The above definitions enable equations A.8, A.9, and A.10 to be written as;-

$$T = \frac{1}{2}\dot{\overline{d}}_g^{\,T}[M]\dot{\overline{d}}_g \tag{A.11}$$

$$P = \frac{1}{2}\bar{d}_g^T [K] \bar{d}_g - \bar{d}_g^T \bar{f} \quad \text{(A.12)}$$

$$R = \frac{1}{2}\dot{\bar{d}}_g^{\,T} [C] \dot{\bar{d}}_g \quad \text{(A.13)}$$

where;-

$$[M] = \text{global mass matrix of structure} = \sum_{e=1}^{E} [M_e]$$

$$[K] = \text{global stiffness matrix of structure} = \sum_{e=1}^{E} [K_e]$$

$$[C] = \text{global damping matrix of structure} = \sum_{e=1}^{E} [C_e]$$

$$\bar{f} = \text{total load vector} = \sum_{e=1}^{E} \left[\bar{f}_{se}(t) + \bar{f}_{be}(t)\right] + \bar{f}_c(t)$$

By substituting equations A.11, A.12, and A.13 into equation A.4, and dropping the brackets around the matrices, produces the resulting equations of motion of a structure in the form;-

$$M\ddot{\bar{d}}_g(t) + C\dot{\bar{d}}_g(t) + K\bar{d}_g(t) = \bar{f}(t) \quad (A.14)$$

Natural damping in a structure is generally extremely small, and thus the damping is often neglected, avoiding the assumption which had to be made to derive equation A.7. In this case, the equations of motion simplify to;-

$$M\ddot{\bar{d}}_g(t) + K\bar{d}_g(t) = \bar{f}(t) \quad (A.15)$$

Equation A.15 represents a system of n coupled second order differential equations, where n is the number of degrees of freedom of the structure. This set of equations can be transformed into a set of n uncoupled second order differential equations by a suitable co-ordinate transformation. A particularly convenient co-ordinate basis for the uncoupled equations is the modal co-ordinate basis, where each equation describes the behaviour of the structure at a particular resonant frequency. This is known as the "normal mode" basis [11],[12], and the transformation can be obtained by solution of the eigenproblem;-

$$M^{-1}K\Phi = \Phi diag(\lambda_i) \quad i = 1, 2, \ldots n \quad (A.16)$$

where Φ is the matrix whose columns ϕ_i are the eigenvectors corresponding to the eigenvalues λ_i of the characteristic matrix $M^{-1}K$. The matrix Φ is known as the modal matrix, and its column vectors ϕ_i describe the modeshapes of the structure. The modal frequencies of the structure ω_i are defined by the eigenvalues as;-

$$\omega_i^2 = \lambda_i \quad i = 1, 2, \ldots n \tag{A.17}$$

Because of reciprocity properties [30],[31], matrices M and K are symmetric, hence it is known that there is a normalisation of Φ (whose columns are only defined up to a multiplicative constant by equation A.16) such that;-

$$\Phi^T M \Phi = I \tag{A.18}$$

where I is the unit matrix of appropriate dimension. Using the above normalisation (equation A.18), it is apparent from equations A.16 and A.17 that;-

$$\Phi^T K \Phi = diag(\omega_i^2) \quad i = 1, 2, \ldots n \tag{A.19}$$

Introducing the co-ordinate transformation (often referred to as the modal transformation) defined by;-

$$\overline{d}_g = \Phi \overline{q} \tag{A.20}$$

where $\overline{q}$ is the vector of modal co-ordinates, equation A.15 can be transformed to;-

$$M\Phi\ddot{\overline{q}}(t) + K\Phi\overline{q}(t) = \overline{f}(t) \tag{A.21}$$

Premultiplying equation A.21 by Φ^T and using equations A.18 and A.19 gives the modal equations;-

$$I\ddot{\overline{q}}(t) + diag(\omega_i^2)\overline{q}(t) = \overline{F}(t) \quad i = 1, 2, \ldots n \tag{A.22}$$

where $\overline{F}(t) = \Phi^T \overline{f}(t)$ is often referred to as the modal force vector, and I is the unit matrix of appropriate dimension.

In general, it is not possible to obtain a co-ordinate transformation which will decouple the damped modal equation (equation A.14), and hence if it is required to include damping terms in the equations, one of two approaches can be followed. The first approach is based on an alternative representation of the C matrix in equation A.14. By assuming that the effect of the damping matrix C is small compared to the effects of the mass matrix M and the stiffness matrix K, then the damping matrix C can be represented as a linear

combination of the mass and stiffness matrices (see [135, Section 6.7.3], [136, Section 4.5.1]), as follows;-

$$C = aM + bK$$

where the scalars a and b can be chosen to best suit the problem in hand. This results in an equation of the form of equation A.15, and so the above decoupling procedure can be applied to produce the modal equations. Then recovering the damping term gives an additional term of the form;-

$$\left[aI + b\, diag(\omega_i^2)\right] \dot{\bar{q}}(t) \quad i = 1, 2, \ldots n$$

Defining a damping ratio ζ_i for mode i as;-

$$\zeta_i = \frac{a + b\omega_i^2}{2\omega_i}$$

gives the following damped form of the modal equations;-

$$I\ddot{\bar{q}}(t) + diag(2\zeta_i\omega_i)\dot{\bar{q}}(t) + diag(\omega_i^2)\bar{q}(t) = \bar{F}(t) \quad i = 1, 2, \ldots n \qquad (A.23)$$

More often, however, it is not possible to even define a C matrix, and so damping is more usually added in a rather heuristic manner simply by adding a term $2\zeta_i\omega_i\dot{\bar{q}}(t)$ to the undamped modal equation (equation A.22), and by choosing nominal values for the ζ_i [136, Section 4.2.2].

The damped modal equations can be conveniently described in state space form for control system analysis and design by choosing the state vector x to represent the modal co-ordinates $\bar{q}$ and their rates $\dot{\bar{q}}$, such that;-

$$x(t) = \begin{pmatrix} \bar{q}(t) \\ \dot{\bar{q}}(t) \end{pmatrix}$$

Hence the state space description can be given by;-

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned}$$

where $x \in \mathcal{R}^{2n}$ is the vector of modal amplitudes and rates, $u \in \mathcal{R}^m$ is the vector of external forces and torques, and $y \in \mathcal{R}^p$ is the vector of nodal displacements and rates. Matrix A is block diagonal, with blocks of the form;-

$$a_i = \begin{pmatrix} 0 & 1 \\ -\omega_i^2 & -2\zeta_i\omega_i \end{pmatrix}$$

and matrix B has corresponding blocks of the form;-

$$b_i = \begin{pmatrix} 0 \\ \Phi_i^T \end{pmatrix}$$

and matrix C also has corresponding blocks, which have the form;-

$$c_i = \begin{pmatrix} \Phi_i & 0 \\ 0 & \Phi_i \end{pmatrix}$$

A.2 Some Theorems Concerning Definite Matrices

Theorem 1 *The n x n matrix A is a definite matrix of rank $r \leq n$* iff *A is real and symmetric and has r positive eigenvalues and $n - r$ eigenvalues equal to zero.*

Proof. Throughout this proof and the next, U will denote a real diagonal matrix of the eigenvalues of A, and X will denote a real orthogonal matrix for which;-

$$A = XUX^T$$

Note that as A is symmetric, a set of n linearly independent eigenvectors can be found even if the eigenvalues are not distinct. (See [137, section 5-16], [138, section 5-3], or [139, section 2-10].) Since an orthogonal matrix is necessarily non-singular, it can be deduced that the rank of U is r, the rank of A. Thus it follows that $n - r$ and only $n - r$ diagonal elements of U are zero and hence A has exactly $n - r$ eigenvalues equal to zero.

If λ is an eigenvalue of A, then it is known that there exists an $x \neq 0$ such that $Ax = \lambda x$, and since $x^T x = \langle x, x \rangle > 0$, it can be deduced that;-

$$\lambda = \frac{x^T A x}{x^T x}$$

If A is definite then $\lambda \geq 0$ and the r non-zero eigenvalues must all be positive.

Conversely, suppose that A has eigenvalues $\lambda_1, \ldots, \lambda_r > 0$ and $\lambda_{r+1} = \ldots = \lambda_n = 0$. Then $A = XUX^T$ implies that the rank of A is r. If the columns of X are eigenvectors $x_1, x_2, \ldots, x_n$ then it is possible to write;-

$$A = \sum_{j=1}^{r} \lambda_j G_j$$

where;-

$$G_j = x_j x_j^T$$

Thus for any non-zero $x \in \mathcal{R}^n$,

$$x^T A x = \sum_{j=1}^{r} \lambda_j (x^T x_j)(x_j^T x)$$

$$= \sum_{j=1}^{r} \lambda_j (x_j^T x)^2 \geq 0$$

Furthermore, if $r < n$, $x_{r+1}^T A x_{r+1} = 0$ then A is therefore non-negative definite. (Sometimes called positive semi-definite.) If $r = n$, then $x_j^T x \neq 0$ for at least one j and so $x^T A x > 0$ and A is positive definite. $\diamond$

Remarks. In particular, it should be noted that the real symmetric matrix A is positive definite *iff* all its eigenvalues are positive.

Theorem 2 *The matrix A is a definite matrix of rank r* iff *there is a definite matrix $A^{1/2}$ of rank r such that $(A^{1/2})^2 = A$.*

Proof. Suppose first that A is definite, then by Theorem 1, the eigenvalues $\lambda_1, \ldots, \lambda_n$ of A are non-negative. Define the matrix $U^{1/2}$ to be $\text{diag}(\lambda_1^{1/2}, \lambda_2^{1/2}, \ldots, \lambda_n^{1/2})$ where the positive square root is chosen in each case, and also define $A^{1/2} = X U^{1/2} X^T$, and thus it can be seen that $A^{1/2}$ is a real symmetric matrix of rank r.

In addition;-

$$(A^{1/2})^2 = X U^{1/2} X^T = X U X^T = A. \quad \diamond$$

Theorem 3 *If A and B are positive definite and non-negative definite matrices, respectively, then the zeros of $det(A\lambda + B)$ are real and non-positive.*

Proof. From Theorem 2, it is known that there exists a positive definite matrix $A^{1/2}$ for which $(A^{1/2})^2 = A$. Using the notation $(A^{1/2})^{-1} = A^{-1/2}$, it is possible to write;-

$$(A\lambda + B) = A^{1/2}(\lambda I + A^{-1/2} B A^{-1/2}) A^{1/2}$$

Putting $C = A^{-1/2}BA^{-1/2}$ and $r = A^{1/2}q$, it can be seen that

$$(A\lambda + B)q = 0$$

iff

$$(\lambda I + C)r = 0$$

The eigenvalues of $A^{-1}B$ are therefore the eigenvalues of the real symmetric matrix C, and conversely, and therefore they are real.

Let λ be an eigenvalue of $A^{-1}B$ with real right eigenvector q. Then $\lambda Aq = -Bq$ and pre-multiplying by q^T gives;-

$$\lambda = -\frac{q^TBq}{q^TAq}$$

Since $q^TAq > 0$ and $q^TBq \geq 0$ it follows that $\lambda \leq 0$. $\diamond$

Remarks. It can be seen from Theorem 3 that if B is also a positive definite matrix, then the zeros of $det(A\lambda + B)$ will be real and negative.

It can also be seen from Theorem 3 that the product of two positive definite matrices will also be a positive definite matrix, and that the product of a positive definite matrix and a non-negative definite matrix will be a non-negative definite matrix.

Theorem 4 *If A, B, C are all positive definite matrices, then the zeros of $det(A\lambda^2 + B\lambda + C)$ have negative real parts.*

Proof. From Theorem 2, matrix A can be factorised such that $A = A^{1/2}A^{1/2}$, and from the remarks following Theorem 3, it is possible to choose matrices D and E which are positive definite, such that;-

$$det(A\lambda^2 + B\lambda + C) = det(A^{1/2}\lambda + D)det(A^{1/2}\lambda + E)$$

From above, and the remarks following Theorem 3, it can be seen that the zeros of $det(A\lambda^2 + B\lambda + C)$ will occur in complex pairs with negative real parts. $\diamond$

Theorem 5 *If $\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \ldots \geq \tilde{\lambda}_{n-r}$ are the eigenvalues of the nth order eigenproblem $Ax = \lambda x$ subject to r constraints of the form;-*

$$a_j^Tx = 0 \quad j = 1, 2, \ldots, n - r$$

then;-

$$\lambda_{j+r} \leq \tilde{\lambda}_j \leq \lambda_j \quad j = 1, 2, \ldots, n-r$$

where $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$ *are the eigenvalues of the unconstrained problem.*

Proof. Throughout this proof and the next, use is made of the Courant-Fischer theorem, which is not proved here due to its length, but a statement of the theorem together with its proof can be found in [138, section 3-6].

From the Courant-Fischer theorem,

$$\lambda_{j+r} \leq max\ R(x) \quad j = 1, 2, \ldots, n-r$$

where $R(x)$ is the Rayleigh quotient (see [138, section 3-1]), and the maximum is over all the non-zero vectors. Also;-

$$max\ R(x) = \tilde{\lambda}_j$$

hence the first inequality $\lambda_{r+j} \leq \tilde{\lambda}_j$.

For the other half of the inequality, consider the eigenvalues of $-A$, denoted by $\mu_1 \geq \mu_2 \geq \ldots \geq \mu_n$, where

$$\mu_p = -\lambda_{n+1-p} \quad p = 1, 2, \ldots, n.$$

Let the eigenvalues of $-A$ under the constraints

$$a_j^T x = 0 \quad j = 1, 2, \ldots, r$$

be $\tilde{\mu}_1 \geq \tilde{\mu}_2 \geq \ldots \geq \tilde{\mu}_{n-r}$.

Then

$$\tilde{\mu}_p = -\tilde{\lambda}_{n+1-r-p} \quad p = 1, 2, \ldots, n-r$$

and applying the above result gives

$$\mu_{r+k} \leq \tilde{\mu}_k \quad k = 1, 2, \ldots, n-r$$

whence $-\lambda_{n+1-r-k} \leq -\tilde{\lambda}_{n+1-r-k}$. Hence writing $j = n+1-r-k$ gives;-

$$\tilde{\lambda}_j \leq \lambda_j \quad j = 1, 2, \ldots, n-r. \quad \diamond$$

Remarks. Theorem 5 implies that the modal frequencies predicted using the "constrained" approach (see Chapter 2) will always be higher than the

actual modal frequencies of the "free" structure. It also implies that the frequencies predicted by a finite element representation will always be higher than the actual frequencies, and that the predicted frequencies will decrease towards the actual frequencies in a monotonic manner as the number of constraints are reduced, that is the number of elements used to represent the structure are increased.

Theorem 6 *If A is a real symmetric matrix with eigenvalues $\lambda_1 \geq \lambda_2 \geq \ldots \geq \lambda_n$, B is a non-negative definite matrix of rank r, $(1 \leq r \leq n)$, and $(A - B)$ has eigenvalues $\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \ldots \geq \tilde{\lambda}_n$, then*

$$\tilde{\lambda}_i \leq \lambda_i \quad i = 1, 2, \ldots, n.$$

Proof. It can be seen that because B is non-negative definite, that

$$x^T(A - B)x = x^T Ax - x^T Bx \leq x^T Ax$$

for any $x \in \mathcal{R}^n$. Thus if $\mathcal{S}$ is any subspace of $\mathcal{R}^n$, then;-

$$max\ x^T(A - B)x \leq max\ x^T Ax$$

where the maxima are over all vectors $x \in \mathcal{S}$ with $x^T x = 1$. In particular, consider all subspaces $\mathcal{B}^{n-i+1}$ of dimension $n - i + 1$ where $1 \leq i \leq n$, then the Courant-Fischer Theorem gives;-

$$\tilde{\lambda}_i = \min_{\mathcal{B}^{n-i+1}} \max_{\substack{x \in \mathcal{B}^{n-i+1} \\ x^T x = 1}} x^T(A - B)x \leq \min_{\mathcal{B}^{n-i+1}} \max_{\substack{x \in \mathcal{B}^{n-i+1} \\ x^T x = 1}} x^T Ax = \lambda_i \quad \diamond$$

Remarks. It can be seen from Theorem 6 that if both A and B are non-negative definite matrices then, in general, $A - B$ is also a non-negative definite matrix. Also, if A is a non-negative definite matrix of rank r, (that is it has $n - r$ zero eigenvalues) and if B is a non-negative definite matrix of rank $n-r$, (that is it has r zero eigenvalues) then with a suitable partitioning of A and B, the matrix $A - B$ can be made positive definite.

The practical implications of this are that for a system described by the equations;-

$$M\ddot{d} + D\dot{d} + Kd = f \tag{A.24}$$

where M is a positive definite matrix, D and K are non-negative definite matrices of rank $n - r$, then if this system is subject to a feedback law of the form;-

$$f = -F_r\dot{d} - F_d d$$

where F_r and F_d are both non-negative definite and of rank greater than or equal to r and can be suitably partitioned, then the matrices of the polynomial defining the closed loop system;-

$$M\ddot{d} + (D + F_r)\dot{d} + (K + F_d)d = 0$$

will all be positive definite, and thus from Theorem 4, the closed loop system will be stable.

Equation A.24 is easily recognised as the equation which describes the dynamics of a flexible structure, and for any real structure matrix M will always be positive definite, and matrices D and K will always be at least non-negative definite. (See Appendix A.1.) Hence stability under these conditions is guaranteed regardless of the actual parameters in the matrices.

These results also imply that the open loop system can only be stabilised if the number of independent actuators is equal to or greater than the number of rigid modes in the system, and that the number of independent sensors is equal to or greater than r. This implies that a necessary condition for stabilisability is that all the rigid modes must be both controllable and observable.

Obviously, if the matrices F_r and F_d are diagonal, these results imply that as long as the diagonal elements are not negative, and that at least the appropriate elements to control any rigid modes are greater than zero, then the closed loop system will be stable.

Appendix B

Computational Methods

B.1 Computation of Eigenvalues and Eigenvectors

In this section the computation of eigenvalues and eigenvectors of real matrices are considered. The eigenanalysis problem can be stated mathematically as finding the eigenvalues λ_i of A and their associated eigenvectors $v_i \neq 0$ which satisfy;-

$$Av_i = \lambda_i v_i$$

The problem of obtaining only the eigenvalues is considerably easier, so this will be considered first. The eigenvalues of the $n \times n$ matrix A can be found as the roots of the equation;-

$$det\,(\lambda I_n - A) = 0$$

However, the direct solution of this equation is not a practical method, thus generally, some version of the well-known QR algorithm [140] is used. The method used here utilised the FORTRAN NAG library routines F01AKF and F02APF (see [111, Volume 4]). The routine F01AKF reduces a real unsymmetric matrix to upper Hessenberg form using stabilised elementary similarity transformations (see [141, pages 339-358]). The routine F02APF calculates all the eigenvalues of an upper Hessenberg matrix using Francis' economical method for performing the QR algorithm without using complex arithmetic. This uses double shifts of origin where the shifts have either two real values or complex conjugate values (see also [141, pages 359-371]).

The computation of eigenvectors as well as eigenvalues can be carried out in a similar manner, and again use has been made of a FORTRAN

NAG library routine, this time F02AGF (see [111, Volume 4]). This routine reduces a real unsymmetric matrix to upper Hessenberg form using stabilised elementary similarity transformations. The eigenvalues and eigenvectors of the upper Hessenberg matrix are computed using the QR algorithm. The eigenvectors of the Hessenberg matrix are transformed back to the original basis via the transformation matrix which was used to obtain the upper Hessenberg form, in order to produce the eigenvectors of the original matrix (see also [141, pages 339-358 and 372-395]).

Both the above procedures have been found to give reliable results, even with numerically "stiff" matrices.

B.2 Computation of Inverse Matrices

In this section the computation of the inverse of a matrix is examined. Initially, the problem of computing the inverse of a real square $n \times n$ matrix A of rank n is considered. This problem can be stated mathematically as obtaining the matrix X such that;-

$$AX = B$$

where B is an $n \times n$ unit matrix, and where the inverse X is usually denoted by A^{-1}. The method of computing X used here is the same as the solution of a set of real linear equations with multiple right hand sides. The FORTRAN NAG library routine F04AEF (see [111, Volume 5]) was utilised and a brief description of its procedure is now given (see also [141, pages 93-110]).

The routine initially decomposes A using Crout's factorisation with partial pivoting, into the form;-

$$PA = LU$$

where P is a permutation matrix, L is lower triangular, and U is unit upper triangular. An approximation to X is found by back-substitution, and the residual matrix R is found, where;-

$$R = B - AX$$

A correction D to X is found by back-substitution in $AD = R$. The estimate of the solution X is replaced by $(X + D)$ and the process repeated until full machine accuracy has been achieved.

Now consider the problem of obtaining an inverse of a real matrix which may not be of full rank, and may not be square, that is A is of order $m \times n$.

Here it is useful to introduce the concept of a "pseudo-inverse" matrix A^+ which satisfies the conditions;-

1. $AA^+A = A$
2. $A^+AA^+ = A^+$
3. $(AA^+)^T = AA^+$
4. $(A^+A)^T = A^+A$

If matrix A is of full rank, then A^+ can be obtained from A as follows;-

$$A^+ = (A^TA)^{-1}A^T \quad \text{for } m \geq n \tag{B.1}$$

or;-

$$A^+ = A^T(AA^T)^{-1} \quad \text{for } m \leq n \tag{B.2}$$

However, from the point of view of numerical computation the singular value decomposition approach is generally better, and this is now described.

A real $m \times n$ matrix A may be factorised by the singular value decomposition as;-

$$A = Q \begin{pmatrix} D \\ 0 \end{pmatrix} P^T \quad \text{if } m \geq n$$

or;-

$$A = Q \begin{pmatrix} D & 0 \end{pmatrix} P^T \quad \text{if } m \leq n$$

where Q is an $m \times m$ orthogonal matrix, P is an $n \times n$ orthogonal matrix, and D is a diagonal matrix of order $k = \min(m, n)$, whose non-negative diagonal elements are the singular values of A.

The $m \times k$ matrix consisting of the first k columns of Q is the left hand singular vector matrix of A, which will be denoted by $\tilde{Q}$. Also, the $n \times k$ matrix consisting of the first k columns of P is the right hand singular vector matrix of A, which will be denoted by $\tilde{P}$. Thus A can be expressed as;-

$$A = \tilde{Q}D\tilde{P}^T$$

By substitution in either equation B.1 or B.2, as appropriate, the pseudo-inverse A^+ of A is then given by;-

$$A^+ = \tilde{P}D^{-1}\tilde{Q}^T \tag{B.3}$$

It should be noted that as matrix D is diagonal, the computation of its inverse is trivial. In order to compute the singular value decomposition of a matrix, use was made of the FORTRAN NAG library routine F02CWF (see [111, Volume 4]), which has the following basic procedure.

The matrix A is first reduced to upper triangular form, by Householder transformations when $m \geq n$, or by Givens plane rotations when $m < n$. The upper triangular form is then reduced to bidiagonal form by Givens plane rotations, and finally the QR algorithm [140] is used to obtain the singular value decomposition of the bidiagonal form.

The computation of the pseudo-inverse matrix is completed by direct implementation of equation B.3 above, noting that the inverse of D is trivial due to its diagonal form.

Note that the equation;-

$$AX = B$$

has a solution for X if, and only if;-

$$(I_m - AA^+)\, B = 0$$

When this condition is satisfied, the general solution for X is given by;-

$$X = A^+B + (I_n - A^+A)\, \epsilon$$

where ϵ is a set of m arbitrary column vectors.

For the case when $m \geq n$ and $rank(A) = n$, then;-

$$A^+A = \left(A^TA\right)^{-1} A^TA = I_n$$

so that;-

$$I_n - A^+A = 0$$

It is therefore clear that in this case, as the existence condition is satisfied, then there is a unique solution for X.

B.3 Numerical Solution of the Algebraic Riccati Equation

Recall from Chapter 5 the various equations defining the linear quadratic regulator problem. That is, the system equations;-

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) \end{aligned}$$

for which a $u(t)$ is sought which minimises the cost function;-

$$\mathcal{V} = \int_0^\infty \left(x^T Q x + u^T R u \right) dt$$

where matrix Q is positive semi-definite, and matrix R is positive definite, and both are given.

It is known (see for example [71],[70]), that the solution to the steady-state problem is given by;-

$$\begin{aligned} u(t) &= -R^{-1} B^T P x(t) \\ &= -F x(t) \end{aligned}$$

where P is an $n \times n$ symmetric matrix which is the positive semi-definite solution of the algebraic Riccati equation;-

$$A^T P + PA - PBR^{-1} B^T P + Q = 0 \qquad (B.4)$$

It can be shown [142], that a unique positive semi-definite solution P exists if, and only if;-

- (A, B) is a stabilisable pair
- (A, C) is a detectable pair

Note also that under these conditions, the matrix $(A - BF)$ will be a stable matrix.

Now let λ be an eigenvalue of the matrix $(A - BF)$, and let ϕ be the corresponding eigenvector. Then;-

$$\begin{aligned} \lambda\phi &= (A - BF)\,\phi \\ &= A\phi - BR^{-1} B^T P\phi \end{aligned} \qquad (B.5)$$

Let $\epsilon = P\phi$, then;-

$$\lambda\phi = A\phi - BR^{-1} B^T \epsilon \qquad (B.6)$$

Note that equation B.5 can be written as;-

$$\lambda P^{-1}\epsilon = A\phi - BR^{-1} B^T P\phi$$

Hence;-

$$\lambda\epsilon = \left(PA - PBR^{-1} B^T P \right) \phi \qquad (B.7)$$

Using the algebraic Riccati equation (equation B.4), equation B.7 can be expressed as;-

$$\begin{aligned} \lambda\epsilon &= \left(-Q - A^T P\right)\phi \\ &= -Q\phi - A^T\epsilon \end{aligned} \tag{B.8}$$

Equations B.6 and B.8 can be written together as;-

$$\lambda \begin{pmatrix} \phi \\ \epsilon \end{pmatrix} = \begin{bmatrix} A & -BR^{-1}B^T \\ -Q & -A^T \end{bmatrix} \begin{pmatrix} \phi \\ \epsilon \end{pmatrix}$$

Hence;-

$$\lambda \begin{pmatrix} \phi \\ \epsilon \end{pmatrix} = H \begin{pmatrix} \phi \\ \epsilon \end{pmatrix} \tag{B.9}$$

where matrix H is known as the Hamiltonian matrix.

Consider now the non-singular tansformation matrix T, given by;-

$$T = \begin{bmatrix} I_n & 0 \\ P & I_n \end{bmatrix}$$

Thus;-

$$\begin{aligned} T^{-1}HT &= \begin{bmatrix} I_n & 0 \\ -P & I_n \end{bmatrix} \begin{bmatrix} A & -BR^{-1}B^T \\ -Q & -A^T \end{bmatrix} \begin{bmatrix} I_n & 0 \\ P & I_n \end{bmatrix} \\ &= \begin{bmatrix} \left(A - BR^{-1}B^T P\right) & -BR^{-1}B^T \\ -\left(A^T P + PA - PBR^{-1}B^T P + Q\right) & -\left(A - BR^{-1}B^T P\right)^T \end{bmatrix} \\ &= \begin{bmatrix} (A - BF) & -BR^{-1}B^T \\ 0 & -(A - BF)^T \end{bmatrix} \end{aligned} \tag{B.10}$$

It is clear from equation B.10 that the eigenvalues of matrix H are the eigenvalues of matrix $(A - BF)$ together with the eigenvalues of the matrix $-(A - BF)^T$. Since a matrix and its transpose have an identical set of eigenvalues, then for every λ that is an eigenvalue of matrix $(A - BF)$, the matrix H has a pair of eigenvalues λ and $-\lambda$.

This property lead to the method of computing the solution of the algebraic Riccati equation based on the eigenanalysis of the Hamiltonian matrix, originally proposed by MacFarlane [143], and Potter [144].

Continuing the analysis, from equation B.9 it is also apparent that for every eigenvalue λ of matrix H, there exists a corresponding eigenvector given by $\begin{pmatrix} \phi \\ \epsilon \end{pmatrix}$, where ϕ is the corresponding eigenvector of matrix $(A-BF)$, and $\epsilon = P\phi$, where matrix P is the solution of the algebraic Riccati equation. Now suppose that the Hamiltonian matrix H has distinct eigenvalues, and let the set of stable eigenvalues be denoted by $\lambda_i, \quad i = 1,\ldots n$. Note that the remaining set of eigenvalues are given by $-\lambda_i, \quad i = 1,\ldots n$.

Let the eigenvectors of matrix H corresponding to the stable eigenvalues λ_i be denoted by $\begin{pmatrix} \phi_i \\ \epsilon_i \end{pmatrix}, \quad i = 1,\ldots n$, and let;-

$$W_1 = \begin{bmatrix} \phi_1 & \phi_2 & \ldots & \phi_n \end{bmatrix}$$

and

$$W_2 = \begin{bmatrix} \epsilon_1 & \epsilon_2 & \ldots & \epsilon_n \end{bmatrix}$$

where the n eigenvectors are chosen such that W_1^{-1} exists. Since ϕ_i and ϵ_i are related by the expression;-

$$\epsilon_i = P\phi_i$$

it can be seen that;-

$$W_2 = PW_1$$

and thus;-

$$P = W_2 W_1^{-1}$$

Note that the assumption of distinct eigenvalues makes the computation of eigenvalues and eigenvectors easier, but the case of non-distinct eigenvalues can also be treated in this manner [145], and with the use of reliable eigenanalysis routines (see earlier), this is not a problem.

B.4 Numerical Solution of the Lyapunov Equation

The need for an algorithm to solve the Lyapunov equation, that is an equation of the form;-

$$A^T P + PA + Q = 0$$

was dictated by the implementation of the "Q-cover" controller reduction algorithm (see Chapter 3).

The problem of obtaining numerical algorithms for the solution of this type of equation has received much attention over the years (see for example [146],[59],[147],[148],[149],[150]), and obviously, a full examination of all the methods would prove to be very time-consuming.

One method which was examined is the algorithm due to Davison and Man [151], which is a recursive algorithm that offers fast solution and low memory requirements. With current digital computers, particularly those with virtual memory operating systems, memory requirements are not generally so much of a problem as with earlier machines, but with the very large matrices associated with flexible structure problems, memory requirements can become excessive.

Full details of the algorithm can be found in [151], but the outline of the algorithm will be given here in the following.

Consider the linear time-invariant system of differential equations;-

$$\dot{x}(t) = Ax(t), \qquad x(0) = x_0 \tag{B.11}$$

where;-

$$Re\,[\lambda_i(A)] < 0, \qquad i = 1, 2, \ldots n$$

and $\lambda_i(A)$ are the eigenvalues of A. Note that this implies that A is a stable matrix. Consider the quadratic form;-

$$V = x^T Q x$$

then;-

$$\dot{V} = -x^T C x \tag{B.12}$$

where C is defined as follows;-

$$A^T Q + QA = -C$$

Note that the (t) symbolism has been dropped for time-dependent variables where time-dependence is implied.

Integrating equation B.12 with respect to time gives;-

$$V(t) = V(0) - \int_0^t x^T C x dt$$

and as $t \to \infty$, $x(t) \to 0$, so that;-

$$x_0^T Q x_0 = \int_0^\infty x^T C x dt \tag{B.13}$$

Consider now the numerical integration of equation B.13, that is;-

$$\int_0^\infty x^T C x dt = \sum_{k=0,1,2,\ldots}^{\infty} h x_k^T C x_k \qquad (B.14)$$

as $h \to 0$. The numerical integration of equation B.11 by the Crank-Nicolson method [152] gives;-

$$x_{k+1} = \left(I - \frac{h}{2}A + \frac{h^2}{12}A^2\right)^{-1} \left(I + \frac{h}{2}A + \frac{h^2}{12}A^2\right) x_k, \quad k = 0, 1, 2, \ldots \qquad (B.15)$$

and on substituting equation B.15 into equation B.14, the following relation is obtained;-

$$x_0^T Q x_0 = h x_0^T \left[C + \Gamma^T C \Gamma + \Gamma^{T^2} C \Gamma^2 + \Gamma^{T^3} C \Gamma^3 + \cdots\right] x_0$$

where;-

$$\Gamma = \left(I - \frac{h}{2}A + \frac{h^2}{12}A^2\right)^{-1} \left(I + \frac{h}{2}A + \frac{h^2}{12}A^2\right)$$

and thus it can be seen that;-

$$Q = h\left[C + \Gamma^T C \Gamma + \Gamma^{T^2} C \Gamma^2 + \Gamma^{T^3} C \Gamma^3 + \cdots\right] \qquad (B.16)$$

Equation B.16 can be written in the following manner;-

$$Q = \lim_{k \to \infty} Q_k \qquad (B.17)$$

where;-

$$Q_{k+1} = \Gamma^{T^{2^k}} Q_k \Gamma^{2^k} + Q_k, \quad k = 0, 1, 2, \ldots \qquad (B.18)$$

with $Q_0 = hC$, and $h \to 0$.

Equations B.17 and B.18 form the basis for the algorithm. Some particular points from [151] worth noting are that the algorithm is numerically stable for all h, and that the algorithm will converge for all h, and that the algorithm will require only $4n^2$ words of memory, and that the computational time is approximately $(2.5k + 4)n^3\mu$, where μ is the multiplication time of the computing machine. Also suggested in [151] is that a suitable choice of h is given by;-

$$h = \frac{1}{200\,|\lambda_{dom}(A)|}$$

where $\lambda_{dom}(A)$ is the dominant eigenvalue of A, that is;-

$$\lambda_i(A) \leq \lambda_{dom}(A), \qquad i = 1, 2, \ldots n$$

Although this algorithm offers speed and low memory requirements, the recursive nature tends to produce rather large errors. The following simple analysis shows how these errors can build up rapidly.

Consider equation B.18 when Q_k is given by;-

$$Q_k = \overline{Q}_k + \Delta_k$$

where $\overline{Q}_k$ is the exact value of Q_k and Δ_k is the error in Q_k, that is;-

$$\begin{aligned} Q_{k+1} &= \Gamma^{T^{2^k}} \left(\overline{Q}_k + \Delta_k\right) \Gamma^{2^k} + \left(\overline{Q}_k + \Delta_k\right) \\ &= \overline{Q}_{k+1} + \left(\Gamma^{T^{2^k}} \Delta_k \Gamma^{2^k} + \Delta_k\right) \\ &= \overline{Q}_{k+1} + \Delta_{k+1} \end{aligned}$$

and thus it can be seen that the errors can easily increase at each iteration.

Indeed, when the implementation of this algorithm was tested on a fourth order problem, the exact solution of which was already known, it was found that although the algorithm converged quite rapidly (11 steps for 6 significant figure accuracy), the solution contained errors of about 5% in each element.

An alternative method for computing the solution to the Lyapunov equation that was also examined is to consider the Lyapunov equation to be a special case of the algebraic Riccati equation. The algebraic Riccati equation is given by;-

$$A^T P + PA - PBR^{-1}B^T P + Q = 0$$

Note that if $R^{-1} = 0$, then the equation becomes;-

$$A^T P + PA + Q = 0$$

which can be recognised as having essentially the same form as the Lyapunov equation.

Consider now the algorithm described in the previous section for the solution of the algebraic Riccati equation. It can be seen that if $R^{-1} = 0$, then the Hamiltonian matrix is given by;-

$$H = \begin{bmatrix} A & 0 \\ -Q & -A^T \end{bmatrix}$$

Using a similar eigenanalysis procedure, it can be seen that the solution to the Lyapunov equation is given by;-

$$P = W_2 W_1^{-1}$$

where W_1 and W_2 are appropriate partitions of the eigenvector matrix of H, as described in the previous section.

Thus the algorithm for computing the solution of the algebraic Riccati equation based on the eigenanalysis of the Hamiltonian matrix can be used to compute the solution of the Lyapunov equation, by setting R^{-1} to zero.

It should be noted that the matrix Q is constrained to be positive semi-definite, hence this algorithm can only be used under this condition. However, this was not found to be a problem in the use of this algorithm for solving the Lyapunov equation as part of the "Q-cover" algorithm (see Chapter 3).

The fact that the solution of a Lyapunov type of equation was required at the start of the "Q-cover" algorithm, and that the remainder of the algorithm was based on this solution, dictated that accuracy was of prime importance, and thus the latter algorithm discussed above was used in preference to the former, because the accuracy of the solution was much better, although this increase in accuracy was obtained at the expense of greater storage requirements, and greater computational time.

B.5 Numerical Computation of State Transition Matrices

The non-homogeneous time-invariant equation describing the dynamics of a time-invariant linear system, known as the state equation, is given by;-

$$\dot{x}(t) = Ax(t) + Bu(t)$$

and its solution is given by;-

$$x(t) = e^{At}x(0) + \int_0^t e^{A(t-\tau)}Bu(\tau)d\tau$$

and this solution equation is known as the state transition equation of a system. For a discrete time system (or a discrete representation of a continuous time system), it is assumed that the inputs are constant over the sampling period T (which will generally be true if the controller is implemented digitally), thus $u(\tau)$ can be taken outside the integral. Also, by considering

the transition of the states over one sample interval, that is from $t = kT$ to $t = [k+1]T$, the solution equation becomes;-

$$x([k+1]T) = e^{AT}x(kT) + \int_{kT}^{[k+1]T} e^{A(T-\tau)}Bu(kT)d\tau$$

The above equation can be written as;-

$$x_{k+1} = \Phi x_k + \Theta u_k$$

which is known as the discrete state equation of a system, where;-

$$\Phi = e^{AT}$$

and

$$\Theta = \int_0^T e^{A(T-\tau)}Bd\tau$$

Obviously, with a suitable choice of the sampling period T, the discrete form of the state equation is a convenient means for simulating a linear time-invariant continuous time system. If this method is to be used, then the computation of the matrices Φ and Θ have to be considered.

There are basically three methods of obtaining the state transition matrices [103, pages 204-206], and these are;-

- Laplace transform method
- Eigenvalue method
- Power series expansion method

The Laplace transform method is not particularly suitable for high order systems, nor is it particularly suitable for implementation on a digital computer. The eigenvalue method can only be used if the A matrix has distinct eigenvalues, so its application is limited. However, the power series expansion method is eminently suitable for implementation on a digital computer, and so is now described.

The power series representation of matrix Φ is given by;-

$$\Phi = e^{AT} = I + AT + \frac{A^2T^2}{2!} + \frac{A^3T^3}{3!} + \cdots$$

Note that the $k\ th$ term of the series is $\frac{A^k T^k}{k!}$ and that the $[k+1]th$ term is $\frac{A^{k+1}T^{k+1}}{[k+1]!}$, thus it can be seen that;-

$$[k+1]th \quad term = \frac{AT}{k+1} \times k\ th \quad term, \qquad k = 0, 1, 2, \ldots$$

Thus this is most convenient for recursive programming. Obviously, additional terms are computed until the result has converged to the desired accuracy.

Note that the forcing function matrix Θ can also be computed along with the Φ matrix, as will now be shown. When matrix B is time-invariant, matrix Θ is given by;-

$$\begin{aligned} \Theta &= B\int_0^T e^{A(T-\tau)}d\tau \\ &= Be^{AT}\int_0^T e^{-A\tau}d\tau \end{aligned}$$

Evaluating the integral gives;-

$$\begin{aligned} \Theta &= -Be^{AT}\frac{1}{A}\left[e^{-A\tau}\right]_0^T \\ &= -Be^{AT}\frac{1}{A}\left[e^{-AT} - I\right] \\ &= \frac{B}{A}\left[e^{AT} - I\right] \\ &= \frac{B}{A}\left[\Phi - I\right] \end{aligned}$$

Expanding matrix Φ as described above, gives;-

$$\begin{aligned} \Theta &= \frac{B}{A}\left[AT + \frac{A^2T^2}{2!} + \frac{A^3T^3}{3!} + \cdots\right] \\ &= B\left[T + \frac{AT^2}{2!} + \frac{A^2T^3}{3!} + \cdots\right] \\ &= B\Gamma \end{aligned}$$

Thus it can be seen that the matrix Γ can also be computed using a recursive programme, in a similar manner as for the Φ matrix described above. In fact, as the computations at each stage are similar, the terms for the matrices Φ and Γ can be computed alongside each other increasing the overall efficiency of the computation.

Appendix C

Controller Design Based on Positive Real Matrices

C.1 Design Method

This design technique has been introduced and discussed in Chapter 5, and here the details of the use of positive real matrices in the design of multivariable control systems is now given, together with a simple scalar example in which bounds on controller gains obtained using this technique are compared with bounds obtained via a classical analysis.

Recall from Chapter 5 the following fundamental definitions;-

1. Square transfer function matrix $G(s)$ is Positive Real if;-
   - $G(s)$ has real elements for real s
   - $G(s)$ has elements which are analytical for $Re[s] > 0$
   - $G^*(s) + G(s)$ is positive semi-definite for $Re[s] > 0$
     (where $G^*(s)$ is the complex conjugate transpose of $G(s)$)
2. Square transfer function matrix $G(s)$ is Strictly Positive Real if;-
   - $G(s)$ has real elements for real s
   - $G(s)$ has elements which are analytical for $Re[s] \geq 0$
   - $G^*(j\omega) + G(j\omega)$ is positive definite for all real ω

Also recall that the system shown in figure C.1 is asymptotically stable in the input/output sense if at least one of the matrices is strictly positive

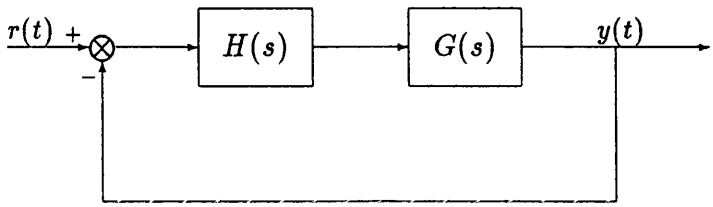


Figure C.1: Schematic of transfer function matrices for Positivity concept

real, and the other is positive real. However, the requirement that the plant matrix $G(s)$ should be at least positive real is rather restrictive, so a technique called "embedding" can be used to introduce new transfer functions without altering the characteristics of the system. Thus the block diagram shown in figure C.1 can be redrawn in the form of figure C.2, and thus to the form in figure C.3, such that;-

$$\tilde{H}(s) = \left(I - HF^{-1}D\right)^{-1} HF^{-1} \quad \text{(C.1)}$$

$$\tilde{G}(s) = FG + D \quad \text{(C.2)}$$

Although this manipulation means that the outputs are no longer equivalent, the characteristic equation for the systems can be shown to be equivalent. That is the characteristic equation for the system shown in figure C.3 is given by;-

$$I + \tilde{H}\tilde{G} = 0 \quad (C.3)$$

Substituting equations C.1 and C.2 into equation C.3 gives;-

$$I - HF^{-1}D + HF^{-1}(FG + D) = 0$$

and thus;-

$$I + HG = 0$$

which is the characteristic equation of the system shown in figure C.1.

It has been shown in [90] that the transfer function matrix representing a structure is always positive real if *ideal* co-located actuators and sensors are used. Real actuators and sensors with finite bandwidth cause the overall tranfer function matrix to cease to be positive real, primarily due to phase

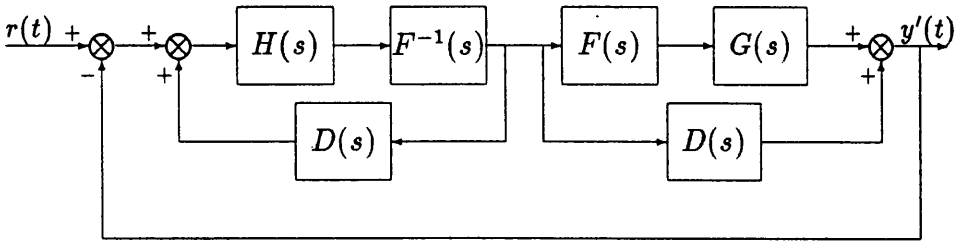


Figure C.2: Schematic of embedded system

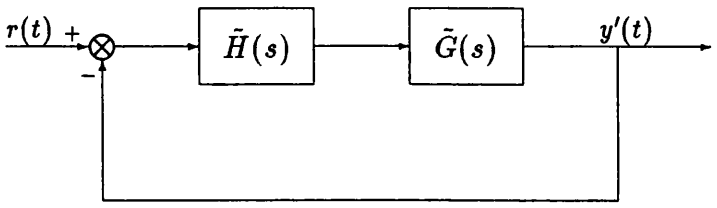


Figure C.3: Schematic of equivalent system after embedding

shifts at higher frequencies. In order for this lack of positivity (or the degree of negativity) to be finite, the structure must possess some structural damping, that is there should be no poles on the imaginary axis in the s plane. This, of course, does pose a problem for any zero frequency modes (rigid modes) of a structure, and thus these have to be approximated, usually by a pair of very low frequency poles. However, it has been found [92] that this type of approximation can cause problems for the subsequent analysis, such that the approximation dominates the analysis at the expense of the rest of the system.

If the tranfer function matrix $G(s)$ is not positive real, than a feedforward term $D(s)$ can be added to make $\tilde{G}(s)$ strictly positive real, and then $H(s)$ can be designed such that $H(s)$ with the feedback $D(s)$, (that is $\tilde{H}(s)$) is positive real. Note that the term $D(s)$ is a purely mathematical entity and thus does not need to be realisable.

This method results in gain stabilisation of the system, so the phase shifts within a loop can exceed 180° without causing instability, and thus accurate phase information at high frequencies is not required.

Thus the design method can be summarised as follows;-

1. Establish positivity/negativity of plant.
   Evaluate smallest eigenvalue of $\frac{1}{2}[G(j\omega) + G^*(j\omega)]$ as a function of ω. That is compute $\delta(\omega)$ against ω.

2. Ensure robustness.
   Plot $\delta(\omega)$ for additional cases of perturbations of the model from nominal, and obtain the *worst case* of negativity.

3. Determine matrix D.
   Select $\delta_0 < \delta_{min}$ where $\delta_{min} = min_{0<\omega<\infty}[\delta(\omega)]$. Then the embedding operator D is given by;-

   $$D = \delta_0 I_{m\times m}$$

   where m is the dimension of $G(s)$. Note that this method of obtaining D results in a constant embedding factor which causes the controller gain to be reduced uniformly over all frequencies. This can be avoided by using a frequency dependent operator, for example;-

   $$D(s) = \delta_0 \frac{(s+a)}{(s+b)}$$

   with $a < b$ (typically $a = 0.1b$), which will preserve a large low frequency gain whilst making $\tilde{G}(s)$ strictly positive real. The parameters a and b must be selected such that $D(s)$ can still effectively cover the largest negative peak in $\delta(\omega)$.

4. Design controller $H(s)$.
   The controller $H(s)$ must now be designed such that the embedded controller $\tilde{H}(s)$ is positive real. If $H(s)$ is a positive definite gain matrix, then embedding $H(s)$ into a positive real $\tilde{H}(s)$ merely places an upper bound on the gains in $H(s)$. When $H(s)$ is a dynamic element, for example, a filter, or a state estimator, then the design becomes trickier. The following procedure can be used to produce a suitable $H(s)$ but it does not produce a unique solution, nor an optimal solution.

- Design a positive real (or strictly positive real) controller H_0 using, for example, a state estimator and state feedback gain matrix.
- Determine the actual controller $H(s)$ by including a negative feedback matrix $D(s)$ around H_0. Since $D(s)$ is strictly positive real, $H(s)$ is ensured to be a stable transfer function matrix by the positivity conditions. Since the embedded controller $\tilde{H}(s)$ is obtained by including a *positive* feedback matrix $D(s)$ around $H(s)$, then $\tilde{H}(s) = H_0(s)$ which has been designed (at least) positive real. Hence the embedded system satisfies the positivity condition, so it is assured to be stable.

This design procedure is now illustrated by application to a simple scalar example.

C.2 Design Example

Consider a plant of the form;-

$$G(s) = \frac{1}{(s+a)(s^2 + 2\zeta\Omega s + \Omega^2)}$$

For this plant;-

$$\begin{aligned} G(j\omega) &= \frac{1}{(j\omega + a)(\Omega^2 - \omega^2 + j2\zeta\Omega\omega)} \\ &= \frac{1}{[a(\Omega^2 - \omega^2) - 2\zeta\Omega\omega^2] + j\,[\omega(\Omega^2 - \omega^2) + 2\zeta\Omega\omega a]} \\ &= \frac{1}{A + jB} \quad \text{(say)} \end{aligned}$$

Note that;-

$$\begin{aligned} \frac{1}{2}\{G(j\omega) + G^*(j\omega)\} &= \frac{1}{2}\left\{\frac{1}{A+jB} + \frac{1}{A-jB}\right\} \\ &= \frac{A}{A^2 + B^2} \\ &= \delta(\omega) \quad \text{(scalar case)} \end{aligned}$$

Hence;-

$$\delta(\omega) = \frac{\omega^2(-a - 2\zeta\Omega) + a\Omega^2}{\omega^6 + \omega^4(a^2 + 4\zeta^2\Omega^2 - 2\Omega^2) + \omega^2(\Omega^4 + 4\zeta^2\Omega^2 a^2 - 2a^2\Omega^2) + a^2\Omega^4}$$

Assuming that $\Omega = 1$, $\zeta = 0.005$, and $a = 3$, gives;-

$$\delta(\omega) = \frac{3 - 3.01\omega^2}{\omega^6 + 7.0001\omega^4 - 16.9991\omega^2 + 9}$$

This function crosses the ω axis at $\omega = 0.998$, and has turning points at $\omega = 0$, $\omega = \pm 0.993$, $\omega = \pm 1.0036$, and $\omega = \pm j1.9996$. At $\omega = 0.993$, $\delta(\omega) = 10.926$, and at $\omega = 1.0036$, $\delta(\omega) = -20.74$. A sketch of $\delta(\omega)$ against ω for $\omega = 0 \rightarrow \infty$ is shown in figure C.4, from which it can be seen that $\delta_0 = 20.74$.

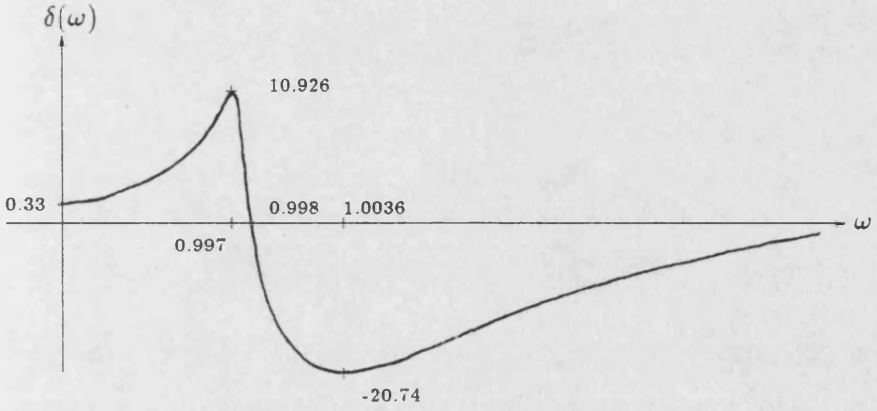


Figure C.4: Sketch of $\delta(\omega)$ against ω

From the positivity theory;-

$$\tilde{H}(s) = (I - HD)^{-1} H$$

(note that in this case, $F = I$) and for stability, $\tilde{H}(s) \geq 0$, hence;-

$$(I - HD)^{-1} H \geq 0$$

For this example, $H = K$, and $D = 20.74$, so;-

$$K \leq \frac{1}{D}$$

Hence;-

$$0 < K \leq 0.0432$$

In order to compare this result with classical stability analysis, examine the characteristic equation of the system, that is;-

$$s^3 + s^2(a + 2\zeta\Omega) + s(2\zeta\Omega a + \Omega^2) + a\Omega^2 + K_{max} = 0$$

With the given data, this becomes;-

$$s^3 + 3.01s^2 + 1.03s + 3 + K_{max} = 0$$

Equating to;-

$$(s + ja)(s - ja)(s + b) = s^3 + bs^2 + a^2s + a^2b = 0$$

gives;-

$$\begin{aligned} a &= 1.015 \\ b &= 3.01 \\ a^2b &= 3.101 \end{aligned}$$

Hence;-

$$K_{max} = 0.101$$

Thus the bounds for stability are;-

$$0 < K \leq 0.101$$

This result shows that the Positivity Concept is conservative, and this has been confirmed in a more detailed study [92], which examined both a single axis model and a three axis model of a two-modal approximation of the Olympus spacecraft.

References

[1] M. Burton and C. Rogers. *Preliminary study of the dynamic control of large spacecraft - Vol. 4 Literature Survey.* Final Report TP 7937, British Aerospace, 1981.

[2] J.N. Aubrun, J.A. Breakwell, and G.J. Chambers. Experimental results for active structural control. In *20th Conference on Decision and Control*, pages 706–709, IEEE, San Diego, CA, December 1981.

[3] D.B. Schaechter. Hardware demonstration of flexible beam control. *Journal of Guidance and Control*, 5(1):48–53, January 1982.

[4] L. Meirovitch, H. Baruh, R.C. Montgomery, and J.P. Williams. Nonlinear natural control of an experimental beam. *Journal of Guidance and Control*, 7(4):437–442, July-August 1984.

[5] J.R. Wertz, editor. *Spacecraft Attitude Determination and Control.* D. Reidel Publishing Company, 1978.

[6] A.L. Greensite. *Analysis and Design of Space Vehicle Flight Control Systems.* Spartan Books (Macmillan and Co. Ltd.), 1970.

[7] M.H. Kaplan. *Modern Spacecraft Dynamics and Control.* John Wiley and Sons, 1976.

[8] C.J.H. Williams, E.B. Crellin, and S.A. Gotts. *Mathematical Methods in Flexible Spacecraft Dynamics.* Final Report ESS/SS 766, British Aerospace, 1976.

[9] P.C. Hughes. *Spacecraft Attitude Dynamics.* John Wiley and Sons, 1986.

[10] F. Janssens. Mathematical modelling of a flexible beam under gravity. *ESA Journal*, 7:195–208, 1983.

[11] L. Meirovitch. *Elements of Vibration Analysis.* Mc.Graw Hill Book Company, London, 1975.

[12] W.T. Thomson. *Theory of Vibration with Applications. 2nd. edn.* Prentice-Hall Inc., Englewood Cliffs, N.J., 1981.

[13] S.D. Stearns. *Digital Signal Analysis.* Haydn Book Company, Inc., 1975.

[14] P.W. Likens and H.K. Bouvier. Attitude control of non-rigid spacecraft. *Astronautics and Aeronautics*, 9:64–71, May 1971.

[15] P.C. Hughes. Attitude dynamics of a three-axis stabilised satellite with a large flexible solar array. *Journal of the Astronautical Sciences*, 20(3):166–189, Nov-Dec 1972.

[16] P.C. Hughes. Dynamics of flexible space vehicles with active attitude control. *Celestial Mechanics*, 9:21–39, 1974.

[17] M. Jamshidi. *Large Scale Systems: Modelling and Control.* Elsevier Science Publishing Co., 1983.

[18] S.V. Salehi. Application of adaptive observers to the control of flexible spacecraft. In *10th Symposium on Automatic Control in Space*, pages Appendix 13–20, IFAC, France, June 1985.

[19] H.B. Hablani. Constrained and unconstrained modes; some modelling aspects of flexible spacecraft. *Journal of Guidance and Control*, 5(2):164–173, March-April 1982.

[20] P.W. Likens, Y. Ohkami, and C. Wong. Appendage modal co-ordinate truncation criteria in hybrid co-ordinate analysis. *Journal of Spacecraft*, 13(10):611–617, Oct 1976.

[21] R.H. MacNeal. *Vibrations of Composite Systems.* Report OSR TN-55-150, Office of Scientific Research, Oct. 1954.

[22] W.W. Hooker and G. Margulies. The dynamical attitude equations for an n-body satellite. *Journal of the Astronautical Sciences*, 12(4):123–128, Winter 1965.

[23] W.W. Hooker. A set of r dynamical equations for an arbitrary satellite having r rotational degrees of freedom. *AIAA Journal*, 8(7):1205–1207, July 1970.

[24] P.W. Likens and G.E. Fleischer. *Large Deformation Modal Coordinates for Non-rigid Vehicle Dynamics.* Technical Report 32-1565, NASA, Nov 1972.

[25] B. Wie, N. Furomoto, A.K. Banerjee, and P.M. Barba. Modelling and simulation of spacecraft solar array deployment. *Journal of Guidance and Control*, 9(5):593–598, Sept-Oct 1986.

[26] S. Levy. Computation of influence coefficients for aircraft structures with discontinuities and sweepback. *Journal of Aeronautical Science*, 14(10):547–560, October 1947.

[27] S. Levy. Influence coefficients of tapered beams computed on SEAC. *Journal of Applied Mechanics*, 20:131–133, March 1953.

[28] H.U. Schuerch. Structural analysis of swept, low aspect ratio, multispar aircraft wings. *Aeronautical Engineering Review*, 11(11):34–41, Nov. 1952.

[29] E.L. Leppert, Jr. An application of IBM machines to the solution of the flutter determinant. *Journal of Aeronautical Science*, 14(3):171–174, March 1947.

[30] S.S. Rao. *The Finite Element Method in Engineering.* Pergamon Press Ltd., Oxford, 1982.

[31] O.C. Zienkiewicz. *The Finite Element Method.* Mc.Graw-Hill Book Company Ltd., 1977.

[32] *NASTRAN Primer.* The MacNeal–Schwendler Corporation, 7442 North Figueroa Street, Los Angeles, California. 90041.

[33] *NASTRAN Theoretical Manual.* The MacNeal–Schwendler Corporation, 7442 North Figueroa Street, Los Angeles, California. 90041.

[34] B.C. Moore. Principal component analysis in linear systems: controllability, observability, and model reduction. *IEEE Transactions on Automatic Control*, AC-26:17–32, 1981.

[35] R.E. Skelton. Cost decomposition of linear systems with application to model reduction. *International Journal of Control*, 32(6):1031–1055, 1980.

[36] R.E. Skelton and P.C. Hughes. Modal cost analysis of linear matrix-second-order systems. *Journal of Dynamic Systems, Measurement and Control*, 102:151–158, September 1980.

[37] S.A. Marshall. The design of reduced-order systems. *International Journal of Control*, 31(4):677–690, 1980.

[38] H.H. Rosenbrock. *State-space and Multivariable Theory.* Thomas Nelson and Sons Ltd., London, 1970.

[39] J.R. Sesak and T. Coradetti. Decentralised control of large space structures via forced singular perturbations. In *AIAA 17th Aerospace Sciences Meeting*, New Orleans, January 1979.

[40] D.A. Wilson. Optimal solution of model reduction problems. *Proceedings IEE*, 117:1161–1165, 1970.

[41] D. Bonvin and D.A. Mellichamp. A unified derivation and critical review of modal approaches to model reduction. *International Journal of Control*, 35:829–848, 1982.

[42] A. Yousuff and R.E. Skelton. An optimal controller reduction by covariance equivalent realizations. *IEEE Transactions on Automatic Control*, AC-31(1):56–60, January 1986.

[43] R.E. Skelton and E.G. Collins. A set of q-Markov covariance equivalent models for discrete-time systems. In *First Symposium on Modelling for Simulation and Control of Lumped and Distributed Parameter Systems*, pages 603–606, Lille, France, 3rd-6th June 1986.

[44] G.S. West-Vukovich, E.J. Davison, and P.C. Hughes. The decentralised control of large flexible space structures. *IEEE Transactions on Automatic Control*, AC-29(10):866–879, Oct 1984.

[45] E.J. Davison and I.J. Ferguson. The design of controllers for the multivariable robust servomechanism problem using parameter optimization methods. *IEEE Transactions on Automatic Control*, AC-26:93–110, January 1981.

[46] E.J. Davison and P. Wong. A robust conjugate-gradient algorithm which minimises l-functions. *Automatica*, 11:297–308, 1975.

[47] W. Gesing and E.J. Davison. An exact penalty function algorithm for solving general constrained parameter optimization problems. *Automatica*, 15:175–188, 1979.

[48] W. Gesing and E.J. Davison. Improvements on a robust conjugate gradient algorithm which minimises l-functions. *Automatica*, 14:515–516, 1978.

[49] D.S. Bernstein and D.C. Hyland. The optimal projection equations for fixed-order dynamic compensation of distributed parameter systems. In *AIAA Dynamic Specialists Conference*, Palm Springs, May 1984.

[50] D.C. Hyland and D.S. Bernstein. Explicit optimality conditions for fixed-order dynamic compensation. In *IEEE Conference on Decision and Control*, San Antonio, Tx., December 1983.

[51] D.S. Hyland and D.S. Bernstein. The optimal projection approach to model reduction and the relationship between the methods of Wilson and Moore. In *23rd Conference on Decision and Control*, Las Vegas, December 1984.

[52] D.S. Hyland. The optimal projection approach to fixed-order compensation: numerical methods and illustrative results. In *AIAA 21st Aerospaces Sciences Meeting*, Reno, NV, January 1983.

[53] D.G. Luenberger. An introduction to observers. *IEEE Transactions on Automatic Control*, AC-16:596–602, 1971.

[54] R.V. Patel and N. Munro. *Multivariable System Theory and Design.* Pergamon Press, 1982.

[55] J. O'Reilly. *Observers for Linear Systems.* Academic Press Inc. (London) Ltd., London, 1983.

[56] M.J. Balas and J.R. Canavin. An active modal control system philosophy for a class of large space structures. In *First Symposium on the Dynamics and Control of Large Spacecraft*, pages 271–285, Blacksberg, Va., June 1977.

[57] D.G. Luenberger. Observers for multivariable systems. *IEEE Transactions on Automatic Control*, AC-11:190–197, April 1966.

[58] R.E. Kalman. A new approach to linear filtering and prediction problems. *Trans. ASME*, 82(D):35 on, 1960.

[59] R.E. Kalman and R.S. Bucy. New results in linear filtering and prediction theory. *Trans. ASME*, Series D:95–108, March 1961.

[60] M.J. Balas. Feedback control of flexible systems. *IEEE Transactions on Automatic Control*, AC-23:673–679, 1978.

[61] F.G. Stremler. *Introduction to Communications Systems.* Addison–Wesley Publishing Company, 1977.

[62] D.M. Wiberg and J.T. Gillis. Estimation of frequencies of vibration using lattices. *IEEE Transactions on Automatic Control*, AC-31(8):790–792, August 1986.

[63] N. Sundararajan and R.C. Montgomery. Adaptive control of a flexible beam using least square lattice filters. *IEEE Transactions on Aerospace and Electronic Systems*, AES-20(5), September 1984.

[64] M.J. Balas. Direct velocity feedback control of large space structures. *Journal of Guidance and Control*, 2(3):252–253, May-June 1979.

[65] A. Arbel and N.K. Gupta. Robust collocated control for large flexible space structures. *Journal of Guidance and Control*, 4(5):480–486, Sept-Oct 1981.

[66] S.M. Joshi and N.J. Groom. Optimal member damper controller design for large space structures. *Journal of Guidance and Control*, 3(4):378–380, July-August 1980.

[67] S.M. Joshi. A class of stable, robust feedback controllers for large space structures. *Acta Astronautica*, 9(5):291–295, 1982.

[68] S.M. Joshi and N.J. Groom. Stability bounds for the control of large space structures. *Journal of Guidance and Control*, 2(4):349–351, July-August 1979.

[69] M.J. Balas. Enhanced modal control of flexible structures via innovations feedthrough. In *2nd Symposium on Dynamics and Control of Large Flexible Spacecraft*, Blacksberg, Va., June 1979.

[70] A.E. Bryson and Y. Ho. *Applied Optimal Control.* Hemisphere Publishing Corporation, 1975.

[71] B.D.O. Anderson and J.B. Moore. *Linear Optimal Control.* Prentice-Hall, Englewood Cliffs, N.J., 1971.

[72] H. Kwackernaak. Asymptotic root loci of multivariable linear optimal regulators. *IEEE Transactions on Automatic Control*, AC-21:378–382, 1976.

[73] J. Sesak and P. Likens. Flexible spacecraft control by model error sensitivity suppression. *Journal of Astronautical Sciences*, 27(2):131–156, Jan 1979.

[74] D.L. Kleinman. On an iterative technique for Riccati equation computation. *IEEE Transactions on Automatic Control*, AC-13:114–115, 1968.

[75] J.D. Roberts. *Linear model reduction and solution of algebraic Ricatti equations by use of the sign function.* Report CUED/B-Control/TR13, University of Cambridge, Department of Engineering, 1971.

[76] T. Kailath. Some new algorithms for recursive estimation in constant linear systems. *IEEE Transactions on Information Theory*, IT-19:750–760, November 1973.

[77] M. Womble and J.E. Potter. A pre-filtering version of the Kalman filter and new numerical integration formulas for Riccati equations. *IEEE Transactions on Automatic Control*, AC-20:378–380, 1975.

[78] M. Jamshidi and F. Bottiger. On the imbedded solutions of algebraic matrix Ricatti equations. In *Proc. Joint Automatic Control Conference*, pages 474–481, U.S.A., 1976.

[79] F.A. Farrar and R.C. Di Pietro. *Comparative evaluation of numerical methods for solving the algebraic matrix Riccati equation.* Technical Report R76-140268-1, United Technologies Research Center, East Hartford, CT, December 1976.

[80] G.A. Hewer and G. Nazaroff. *A survey of numerical methods for the solution of algebraic Riccati equations.* Technical Report, Naval Weapons Center, China Lake, CA.

[81] R.E. Lindberg and R.W. Longman. On the number and placement of actuators for independent modal space control. *Journal of Guidance and Control*, 7(2):, March-April 1984.

[82] B. Porter and T.R. Crossley. *Modal Control.* Taylor and Francis Ltd., London, 1972.

[83] J.D. Simon and S.K. Mitter. A theory of modal control. *Information and Control*, 13:316–353, 1968.

[84] B. Francis, O.A. Sebakhy, and W.M. Wonham. Synthesis of multivariable regulators: the internal model principle. *Journal of Appl. Math. Optimiz.*, 1:64–86, 1974.

[85] N. Wiener. *The Interpolation and Smoothing of Stationary Time Series.* M.I.T. Press, Cambridge, Massachusetts, 1949.

[86] H.H. Rosenbrock. *Computer Aided Control System Design.* Academic Press, London, 1974.

[87] D.J. Hawkins. Pseudodiagonalisation and the Inverse Nyquist Array method. *Proceedings IEE*, 119:337–342, 1972.

[88] A.G.J. MacFarlane. *Complex Variable Methods for Linear Multivariable Feedback Systems.* Taylor and Francis Ltd., London, 1980.

[89] A.G.J. MacFarlane and J.J. Belletrutti. The Characteristic Locus design method. *Automatica*, 9:575–588, 1973.

[90] R.J. Benhabib, R.P. Iwens, and R.L. Jackson. Stability of distributed control for large flexible structures using positivity concepts. *AIAA Guidance and Control Conference*, (79-1780), 1979.

[91] R.P. Iwens, R.J. Benhabib, and R.L. Jackson. A unified approach to the design of large space structure control systems. *JACC*, 1980.

[92] R.D. Hocken. *A preliminary study into the design of controllers for flexible spacecraft using positive real matrices.* Technical Report, British Aerospace, 1983.

[93] S. Timoshenko. *Theory of Plates and Shells.* Mc.Graw-Hill Book Company, Inc., London, 1940.

[94] P. Hammond. *Electromagnetism for Engineers.* Pergamon Press, 1978.

[95] B.S. Massey. *Mechanics of Fluids.* Van Nostrand Reinhold, 1979.

[96] B.A. White, R.T. Lipczynski, and A.R. Daniels. Real-time control of a linear synchronous motor for use in advanced ground transport. In *Real-Time Control of Electromechanical Systems, IERE Publication, No.58*, pages 49–52, March 1984.

[97] W.I. Fletcher. *An Engineering Approach to Digital Design.* Prentice-Hall Inc., 1980.

[98] L.A. Dale. *Real-time modelling of multi-machine power systems.* PhD thesis, University of Bath, School of Electrical Engineering, 1986.

[99] *MC68000 16/32 Bit Microprocessor Programmers Reference Manual.* Motorola Inc., 1984.

[100] M. Richards, A.R. Aylward, P. Bond, R.D. Evans, and B.J. Knight. TRIPOS—a portable operating system for minicomputers. *Software Practice and Experience (GB)*, 9(7):513–526, July 1979.

[101] B.A. Bowen and R.J.A. Buhr. *The Logical Design of Multiple-microprocessor Systems.* Prentice Hall.

[102] F.R.A. Hopgood, D.A. Duce, J.R. Gallop, and D.C. Sutcliffe. *Introduction to the Graphical Kernel System (GKS).* Academic Press, London, 1983.

[103] B.C. Kuo. *Digital Control Systems.* Holt, Rinehart and Winston, Inc., 1980.

[104] T.J. King. *TRIPOS User Guide.* School of Mathematics, University of Bath, 1983.

[105] T.J. King. *TRIPOS Programming Guide.* School of Mathematics, University of Bath, 1983.

[106] T.J. King. *TRIPOS Technical Guide.* School of Mathematics, University of Bath, 1983.

[107] M. Richards and C. Whitby-Strevens. *BCPL — the Language and its Compiler.* Cambridge University Press, 1980.

[108] F. da Cruz. *KERMIT protocol manual, 5th Edition.* Columbia University Center for Computing Activities, New York, New York 10027, 1984.

[109] *GINO-F User Manual.* CADCentre Ltd., High Cross, Madingley Road, Cambridge., October 1983.

[110] *GINOGRAF User Manual.* CADCentre Ltd., High Cross, Madingley Road, Cambridge., April 1976.

[111] *NAG Library Mark 11.* Numerical Algorithms Group, NAG Central Office, Mayfield House, 256 Banbury Road, Oxford UK., 1984.

[112] E.J. Davison and S.H. Wang. An algorithm for the calculation of transmission zeros of the system (C,A,B,D) using high gain output feedback. *IEEE Transactions on Automatic Control*, AC-23:738–741, August 1978.

[113] M.G. Safonov. *Stability and Robustness of Multivariable Feedback Systems.* M.I.T. Press, Cambridge, MA., 1980.

[114] R.B. Bate, D.D. Mueller, and J.E. White. *Fundamentals of Astrodynamics.* Dover Publications Inc., New York, 1971.

[115] N.K. Gupta, M.G. Lyons, J.N. Aubrun, and G. Margulies. Modelling, control, and system identification methods for flexible spacecraft. *AGARD*, (260):12.1–12.41, 1981.

[116] G. Thieme. Attitude control of large flexible spacecraft. *AGARD*, (350):18.1–18.15, Sept 1983.

[117] J.G. Lin, D.R. Hegg, Y.H. Lin, and J.E. Keat. Output feedback control of large space structures: an investigation of four design methods. In *Second Symposium on the Dynamics and Control of Large Flexible Spacecraft*, Blacksberg, Va., June 1979.

[118] J.H. Lala. Fault detection, isolation, and reconfiguration in the fault tolerant Multiprocessor. *Journal of Guidance and Control*, 9(5):585–592, Sept-Oct 1986.

[119] P.C. Hughes and T.M. Abdel-Rahman. Stability of proportional-plus-derivative-plus-integral control of flexible spacecraft. *Journal of Guidance and Control*, 2(6):499–503, Nov-Dec 1979.

[120] E.D. Scott. Super mode rejection technique and complex variable bending mode representation. *AIAA Journal*, (80-1793):, 1980.

[121] S.J. Dodds and S.E. Williamson. A signed switching time bang-bang attitude control law for fine pointing of flexible spacecraft. *International Journal of Control*, 40(4):795–811, 1984.

[122] M.J. Balas and C.R. Johnson. Towards adaptive control of large structures in space. In *Workshop on Applications of Adaptive Control*, 1980.

[123] G. Prado and R.K. Pearson. Efficient techniques for system identification of large space structures. In *Joint Automatic Control Conference*, San Francisco, 1980.

[124] F.E. Thau and R.C. Montgomery. Adaptive/learning control of large space structures: system identification techniques. In *Joint Automatic Control Conference*, San Francisco, 1980.

[125] R.J. Benhabib and F.C. Tung. Large space structures control: system identification versus direct adaptive control. In *Joint Automatic Control Conference*, San Francisco, 1980.

[126] D.A. Johnson. Spacecraft identification through Kalman filtering. In *Second Symposium on Dynamics and Control of Large Flexible Spacecraft*, Blacksberg, Va., June 1979.

[127] C. Weeks. The control and estimation of large space structures. In *Joint Automatic Control Conference*, San Francisco, 1980.

[128] D.H. Owens and A. Chotai. On eigenvalues, eigenvectors and singular values in robust stability analysis. *International Journal of Control*, 40(2):285–296, 1984.

[129] A. Herrera-Vaillard, J. Paduano, and D. Downing. Sensitivity analysis of automatic flight control systems using singular-value concepts. *Journal of Guidance and Control*, 9(6):621–626, Nov-Dec 1986.

[130] G. Schulz. Low authority control of flexible spacecraft via numerical optimization. *AGARD*, (350):19.1–19.16, 1983.

[131] T.W.C. Williams. Numerically reliable software for control: the SLICE library. *IEE Proceedings Part D*, 133(2):72–82, March 1986.

[132] H.B. Hablani. A more accurate modelling of the effects of actuators in large space structures. *Acta Astronautica*, 8:361–376, 1981.

[133] R.J. Haftka. Optimum placement of controls for static deformation of space structures. *AIAA Journal*, 22(9):1293–1298, Sept 1984.

[134] Y. Ohkami, O. Okamoto, and T. Kida. Simulation of a digital controller for flexible spacecraft control. In *Second Symposium on the Dynamics and Control of Large Flexible Spacecraft*, Blacksberg, Va., June 1979.

[135] K.H. Huebner and E.A. Thornton. *The Finite Element Method for Engineers.* John Wiley and Sons, 1982.

[136] G.B. Warburton. The influence of the finite element method on developments in structural dynamics. In O.C. Zienkiewicz, E.Y. Rodin, and R. Glowinski, editors, *Energy Methods in Finite Element Analysis*, chapter 4, John Wiley and Sons, 1979.

[137] W. Kaplan. *Advanced Mathematics for Engineers.* Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1981.

[138] S. Perlis. *Theory of Matrices.* Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1952.

[139] P. Lancaster. *Theory of Matrices.* Academic Press, Inc. (London) Ltd., London, 1969.

[140] J.H. Wilkinson. *The Algebraic Eigenvalue Problem.* Oxford University Press, London, 1965.

[141] J.H. Wilkinson and C. Reinsch. *Handbook for Automatic Computation. Volume 2, Linear Algebra.* Springer–Verlag, 1971.

[142] W.M. Wonham. *Linear Multivariable Control: A Geometric Approach.* Springer–Verlag, New York, 1979.

[143] A.G.J. MacFarlane. An eigenvector solution of the optimal linear regulator. *Journal of Electronic Control*, 14:643–654, June 1963.

[144] J.E. Potter. Matrix quadratic solutions. *SIAM Journal of Applied Mathematics*, 14:496–501, May 1966.

[145] K. Martensson. On the matrix Riccati equation. *Inf. Sci.*, 3:17–49, 1971.

[146] R.E. Kalman and J.E. Bertram. Control system analysis and design via the second method of Lyapunov. *Trans. ASME*, Series D:371–393, June 1960.

[147] S. Barnett and C. Storey. Solution of the Lyapunov matrix equation. *Electronics Letters*, 2:466–467, December 1966.

[148] S. Barnett and C. Storey. The Lyapunov matrix equation and Schwarz's form. *IEEE Transactions on Automatic Control*, AC-12:117–118, February 1967.

[149] S. Barnett and C. Storey. Remarks on numerical solution of the Lyapunov matrix equation. *Electronics Letters*, 3:417–418, September 1967.

[150] H.M. Power. Solution of the Lyapunov matrix equation with a diagonal input matrix obtained without matrix inversion. *Electronics Letters*, 3:325–326, July 1967.

[151] E.J. Davison and F.T. Man. The numerical solution of $A^TQ + QA = -C$. *IEEE Transactions on Automatic Control*, AC-13:448–449, August 1968.

[152] E.J. Davison. A high order Crank-Nicolson technique for solving differential equations. *Computer Journal*, 10:195–197, August 1967.

Publications

The following pages contain a copy of a paper resulting from the work described herein presented by the author at the IFAC-IMACS conference entitled "First Symposium on Modelling and Simulation for Control of Lumped and Distributed Parameter Systems", held in Lille, France, 3rd–6th June 1986.

Modelling and Control of a Simple Flexible Satellite Model

T.D.Wood
British Aerospace Dynamics Group,
Space & Communications Div'n,Bristol, U.K. BS99 7AR.

B.A.White
Royal Military College of Science,
Shrivenham,Wiltshire, U.K. SN6 8LA.

Abstract

Most studies of the problems of modelling and control of flexible satellites have been carried out using eigenvalue analysis or simplified simulation exercises. This paper describes an experimental rig which is used for verifying mathematical models of flexible satellites, and for investigating control systems for flexible satellites to real-time application level.

1 Introduction

This paper describes an experimental rig developed to investigate aspects of modelling and control of large flexible space structures.

Numerous papers have been published on the subject of modelling and control of large flexible space structures but those dealing with 'modern' control techniques have almost exclusively drawn conclusions about these techniques via eigenvalue analysis, or simple simulation exercises in which a reduced-order model is used to describe the plant, and where system states are assumed to be available, which in practise they are not. (For example, see references 1,2,3,4). Due to the high-order nature of a flexible spacecraft model, the design of state estimators is not a trivial task, generally this is accomplished in the same manner as the control law itself, i.e. via a reduced-order model. However, control systems designed using reduced-order models rarely perform as expected when applied to the 'real' system due to effects of the unmodelled dynamics.

Other frequently neglected aspects of spacecraft models include actuator and sensor dynamics and non-linearities.

Some notable exceptions are the experimental rigs such as those discussed in references 5 and 6, both located in the U.S.A.

The purpose of the experimental rig described herein is to allow the complete investigation of control systems (primarily for attitude control but also for vibration control) for flexible spacecraft to real-time application level.

This paper is only concerned with the details of the experimental rig, results of work carried out using the rig will be presented on another occasion.

2 Mechanical Hardware

2.1 The Beam

The structure can be considered as a long thin beam of aluminium alloy with approximate dimensions of 2.0m. x 0.15m. x 0.003m. (see figure 1). It is supported by a low friction pivot at its midpoint with the pivotal axis in the vertical plane. Positional control of the structure about this axis is a minimal objective of all control systems which are applied to it, representing single-axis attitude control.

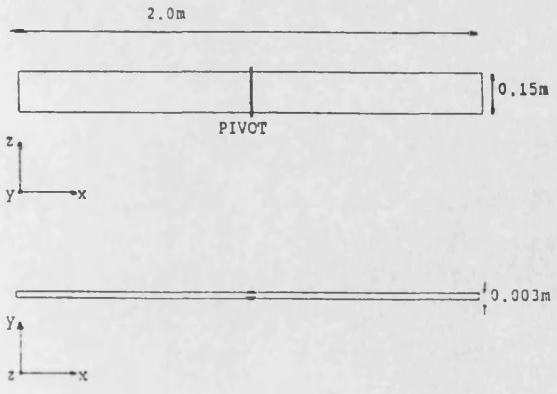


Figure 1 General layout of beam

The beam has holes of approximately 0.03m. diameter at intervals of 0.1m. along its length, which are used to mount mass reaction actuators (see section 2.2), or loading masses. The loading masses are weights of approximately 0.8 Kg. which can be used to change the mass distribution of the beam, hence altering its dynamic characteristics, in order to investigate the sensitivity of the control systems to parameter variation.

The vibration characteristics of the beam with various loading configurations were obtained using a finite element model. The first five natural frequencies of the beam for the case of the beam with no additional masses, and a case of uneven mass distribution are shown in tables 1 and 2, respectively.

Rad/s	Hz
0.00	0.00
3.22	1.31
42.11	6.70
73.30	11.67
142.40	22.66

Table 1 Frequencies of five lowest modes of unloaded beam.

Rad/s	Hz
0.00	0.00
6.72	1.07
24.83	3.95
41.66	6.63
79.86	12.71

Table 2 Frequencies of five lowest modes of symmetrically loaded beam.

2.2 Actuators

A torque actuator is mounted coaxially with the support pivot of the beam which provides torques about the pivotal axis. This actuator is based on the reaction wheel principle where a torque is created as a reaction to the acceleration of an inertia wheel by a d.c. motor.

In addition to the central torque actuator, electromagnetic mass reaction actuators can be mounted at any of the mounting hole locations on the beam. These actuators are based on a reaction force from the acceleration of a steel shuttle in an electromagnetic field. They do have one major limitation in that they have to be driven by a signal with a zero mean otherwise the actuator shuttle is pushed beyond its operating region where the actuator no longer produces a force. Consequently, these actuators are only used for vibration suppression and thus are not used for position control.

2.3 Sensors

The position of the structure about its pivotal axis is measured using a high-resolution servo potentiometer.

The displacements of the structure in the horizontal plane (only displacements in the y direction are of interest, see figure 1) at various points of the structure are measured using ultrasonic position transducers, originally developed at the University of Bath for application to a magnetically-levitated vehicle.[7] These are based on determining the phase shift between the transmitted signal and the received signal which is reflected by the beam.

For model or simulation verification, a bank of these transducers is used to obtain information at a large number of points on the structure, but only a small subset of these are used for feedback purposes, generally only one or two.

3 Electronic Hardware

3.1 Digital

The heart of the rig is a dedicated multiprocessor computing system, comprised of three single-board computers (SBC's) arranged as one master and two slaves. Each SBC is constructed around the Motorola M68000 16/32 bit microprocessor[8] which has a 24 bit address bus, 16 bit external data bus, and 32 bit internal registers. The SBC's were developed at the University of Bath particularly for multiprocessor configuration for application to real-time simulation and control problems.

Each SBC has the capability to accommodate an 8,10, or 12MHz CPU, an appropriate MMU with parallel ports, a floppy-disc controller, two serial ports (RS 232 standard), and either 1/4 or 1 Mbyte of dynamic RAM.

In this application, the slave SBC's have 1/4 Mbyte of on-board dynamic RAM, whereas the master SBC has 1 Mbyte of on-board dynamic RAM. Also, only the master SBC is furnished with a MMU and a floppy-disc controller. One serial port of the master SBC is connected to a terminal, and the other can be connected to the College's mainframe facility.

The SBC's busses are connected to a common backplane bus, onto which is also connected a 1/2 Mbyte dynamic RAM card, a graphics card driving a high-resolution colour graphics monitor, a ADC/DAC card, a digital signal processing card for the ultrasonic position transducers, and a bus arbitration card. (See figure 2).

The 24 bit address bus of the M68000 CPU enables it to directly access up to 16 Mbytes of storage, so the system is arranged such that any memory location in the system can be accessed by any SBC. Thus, any SBC can access not only it's own on-board memory, but also the memory of any other SBC, or can access the 1/2 Mbyte backplane memory, or the graphics card, or the I/O cards, giving great flexibility to the programmer.

The bus arbitration card is a state machine[9] which ensures that requests for the use of the backplane bus by the SBC's are controlled so that only one SBC can access the backplane bus at any one time, thus avoiding the possibly catastrophic 'deadly embrace' should two SBC's access the bus simultaneously. When a SBC wants to use the backplane bus it issues a 'bus request' signal to the arbitration card, and waits for a 'bus grant acknowledge' signal to be returned from the arbitration card. The arbitration card checks the priority level of any pending 'bus request' signals and issues a 'bus grant acknowledge' signal to the waiting SBC with the highest priority. When a SBC has finished using the backplane bus, the arbitration card again checks for any pending 'bus request' signals, and issues a 'bus grant acknowledge' signal, if appropriate.

The master SBC is connected to a 40 Mbyte Winchester disc, and the floppy-disc controller controls a pair of 5-1/4" flexible disc drives, each quad-density, double-sided disc having about 800 Kbytes of storage space.

The two I/O cards connected to the backplane, as already mentioned, are a ADC/DAC card, and a digital signal processing card for the ultrasonic position transducers. The ADC/DAC card comprises a high-speed 'sample and hold',and a 12 bit ADC for the one analogue input, and three high-speed 12 bit DAC's for the three analogue outputs. The analogue input originates from the analogue signal processing rack, and the the three analogue outputs from the DAC's are fed to the analogue processing rack for appropriate processing to drive the actuators. The digital signal processing card is comprised of the oscillator and divider circuits for the various reference signals, and a state machine[9] for each transducer. The output of each state machine is a 16 bit word representing the displacement measured by each transducer.

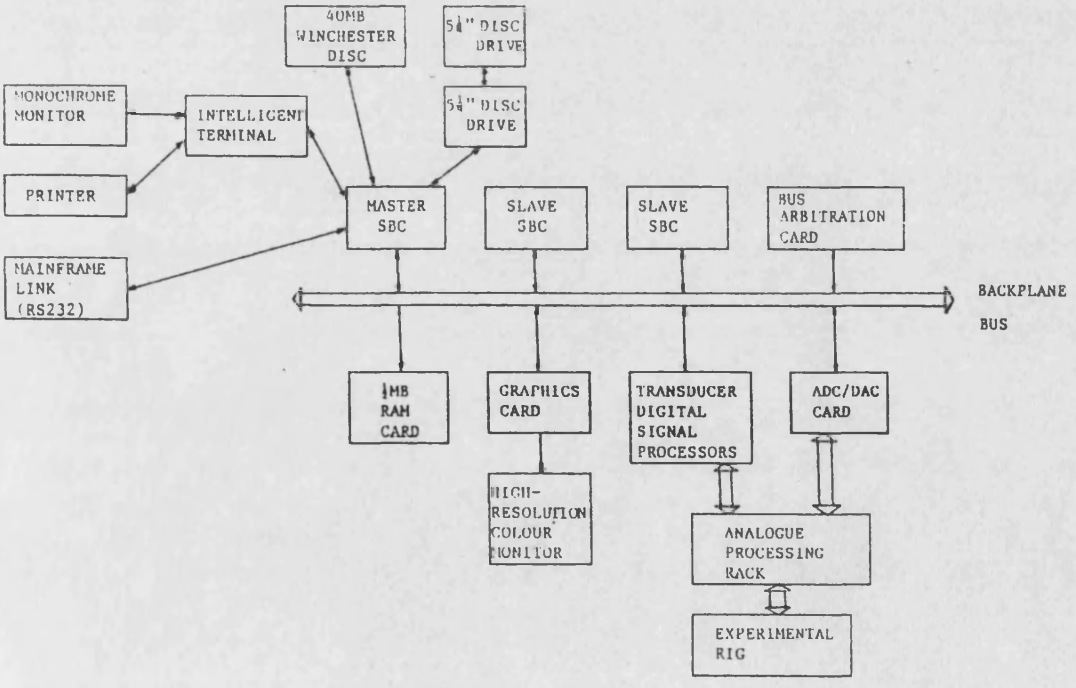


Figure 2 Hardware configuration

The firmware incorporated in the intelligent terminal supports a high resolution graphics display as well as the usual alphanumeric display, enabling graphs to be plotted on the terminal screen. This avoids the need to switch the colour graphics monitor between the real-time display and a graph-plotting display during simulation or real-time control exercises. This also enables hard copies of graphs to be obtained on a printer connected to the terminal's parallel port via a 'screen dump' facility.

3.2 Analogue

The analogue processing rack contains the power amplifiers necessary to drive the actuators from the analogue signals from the DAC's, and the discrete component ultrasonic transducer receiver head amplifiers.

The power amplifiers are basically voltage to current converters, which produce a level of current in either of the two coils of an actuator depending on the voltage presented at their inputs. The power amplifier for the reaction wheel actuator produces an acceleration of the drive motor which is dependent on the voltage at its input.

4 Software

4.1 Operating System

The operating system for this computing system is a multiprocessor extension of the TRIPOS multi-tasking operating system,[10] which was also developed at the University of Bath. Code running on slave SBC's appears to the operating system as additional tasks, but execution of these 'slave' tasks is independent of execution of tasks on the master SBC, or of other 'slave' tasks, except for inter-task communication where specified by the programmer.

Inter-task communication is facilitated by passing 'packets' between tasks. A packet contains five fields, a link field, an identification field, a type field, a result field, and an argument field. The link field provides a pointer to the next packet on a task's packet queue, and the identification field contains a parameter identifying the task which the packet is bound for, but on passing a packet, the identification field is altered to the sending task's i.d. so that the packet can be returned if so desired. The type field contains a parameter which enables the receiving task to identify what it should do with the packet, and likewise for the sending task if the packet is returned. The result field contains two parameters which are only set by the receiving task before the return of a packet to enable the receiving task to return results of processing the packet to the sending task. The argument field contains up to seven parameters which are used by the sending task to pass information to the receiving task for processing.

Each task has a priority associated with it which enables the programmer to ensure that a task which should be executed as rapidly as possible has highest priority so that it is executed whenever possible. The operating system has a task scheduler which attempts to run the task with the highest priority which is available to be run at all times. Thus when a task is suspended at any time, the scheduler attempts to identify the task with the highest priority which is waiting to be run, and runs that task until it too suspends, when it again looks for the waiting task with the highest priority, and runs that task until it suspends, and so on, ad infinitum.

This is obviously a highly desirable feature of a system which is to be used for real-time control, so that the code which evalnates the control law is executed as rapidly as possible with minimal interuption, while other less important code, such as data-logging, are executed when time is available.

The operating system is written in BCPL[11] (except for the machine dependent parts which are written in M68000 assembly code), which is a typeless, block-structured language, with a floating-point extension. Due to the ease in which operating system functions can be called, (such as the packet passing commands) BCPL is also used for the simulation and data-processing software.

4.2 Applications

As an example of how the software can be structured using tasks, consider the simulation of a system subject to a state feedback law, where the states are obtained via a state estimator. (A particularly relevant example for control of flexible spacecraft) The problem can be organised such that the master SBC runs three tasks, (in addition to the operating system tasks)

1. A monitor task that provides the user interface to the simulation and is the only task which the user communicates directly with, and which offers various functions such as allowing parameter or input level changes, or logging, displaying, and storing of simulation variables.

2. A simulation control task that basically controls the computations which are carried out on the slave SBC's, and ensures synchronism of the computations, and can produce timing information for real-time simulations.

3. A real-time display task that provides a real-time display on the high-resolution colour graphics monitor of simulation variables. The display can be controlled by the user via the monitor task.

As already mentioned, the computational burden of the simulation is carried by the slave SBC's, and in this case, can easily be split such that one slave SBC computes the plant states and outputs, and the other computes the estimated states and outputs, with the feedback signal computation being split between them. (See figure 3 for a graphical description of this structure.)

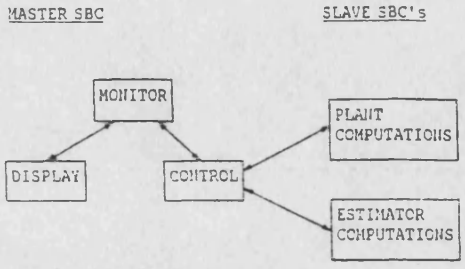


Figure 3 Task structure for plant/estimator simulation

With this type of structure, the simulation runs continuously, thus it is possible for the monitor task to be displaying time histories, for example, of a previous simulation under user control, while another simulation condition is being run.

In addition to the simulation and control implementation facilities, the rig is also used for model verification studies. Common practise for modelling large flexible structures (e.g. reference 12) is to describe the structure using a finite-element model of the form:-

$$Md'' + Kd = f$$

where d is the vector of displacements at element end-points, f is the vector of applied forces at those points, M is a matrix describing the mass properties of the elements, and K is a matrix describing the stiffness properties of the elements. (The prime indicates differentiation with respect to time.) The modal form of these equations is then obtained via a co-ordinate transformation so that the equations can be written as a set of second-order equations in the form:-

$$Iq'' + Wq = F$$

where q is the vector of generalised co-ordinates (sometimes called modal co-ordinates), F is the vector of generalised forces at these co-ordinates, I is a unit matrix of appropriate dimension, and W is a diagonal matrix where each element is the sqnare of the natural frequency of each mode.

Generally, the full set of modal equations produces a model of much higher order than can be tolerated for control system design purposes, so a reduced-order model is chosen, frequently as a subset of the modal equations using various selection criteria.[13,14,15] The rig is used to examine the validity of these reduced-order models by comparing the behavior of the models with the real structure. Also, parameters of the modal equations are identified from the real structure using Fourier analysis techniques, allowing the validity and accuracy of the finite-element models to be verified.

5 Mainframe Link

The link to the mainframe computing facilities enables data to be transferred between the rig and the College mainframes. Thus control system design algorithms which are implemented on the mainframes can be used to design feedback systems, and then the resulting feedback system data can be easily transferred to the rig computing system where the feedback design can be rapidly simulated. If the simulations show promising results, it is then a relatively simple matter to set up the control system and test it on the real structure.

6 Conclusions

The experimental rig described here is being used to validate conclusions based on simulation studies of control systems applied to flexible spacecraft. Although the structure is not as complex as a real large flexible spacecraft, it contains sufficient similarities to enable useful conclusions to be formed, without the constraints of simplifying assumptions used in most simulation studies of flexible spacecraft control. In addition, it also forces an examination of the practicalities of any proposed control system, an aspect which is not always prevalent in many simulation studies.

The facility to carry out simulations using the same computing system that is used for the real-time control implementation is immensely useful as software developed for simulations can be used for implementation with little modification.

7 Further Work

It is hoped to extend the structure at a later date to a two-dimensional model to enable the analysis of two-dimensional satellite systems with cross-coupling between axes, and to make improvements to various aspects of the structure, such as the use of an air bearing to further reduce friction at the support. Also, the use of compressed air jets as actuators is being investigated to overcome the drawbacks of the electromagnetic mass reaction actuators, and because of their similarity to reaction jets which are currently used on many spacecraft. This will enable the investigation of the behavior of control systems which use these very non-linear actuators.

It is also anticipated that the SBC's will be replaced with newer versions based on the Motorola M68020, the full 32 bit CPU, and it's floating-point co-processor, the M68881, which will greatly increase the floating-point computational power of the computing system, allowing more sophisticated control schemes to be investigated in real-time.

8 Acknowledgements

This work was supported by an S.E.R.C. Case award in collaboration with British Aerospace plc., Dynamics Group. The authors grateful acknowledge the use of the NASTRAN finite-element analysis package at B.Ae. plc., Bristol.

9 References

1. Balas,M.J.
"Direct output feedback control of large space structures."
Journal of the Astronautical Sciences, Vol.XXVII,No.2,April-June 1979, pp157-180.

2. Sesak,J.R., & Coradetti,T.
"Decentralized control of large space structures via forced singular perturbations."
17th Aerospaces Sciences Meeting, New Orleans, La. Jan. 1979. AIAA 79-0195.

3. Dodds,S.J., & Williamson,S.E.
"A signed switching time bang-bang attitude control law for fine pointing of flexible spacecraft."
International Journal of Control, Vol.40,No.4,1984, pp795-811.

4. Meirovitch,L., Baruh,H., & Oz,H.
"A comparison of control techniques for large flexible systems."
Journal of Guidance and Control, Vol.6,No.4,July-Aug.1983, pp302-310.

5. Schaechter,D.B.
"Hardware demonstration of flexible beam control."
Journal of Guidance and Control, Vol.5,No.1,Jan.1982, pp48-53.

6. Meirovitch,L., Baruh,H., Montgomery,R.C., & Williams,J.P.
"Nonlinear natural control of an experimental beam."
Journal of Guidance and Control, Vol.7,No.4,July-Aug.1984, pp437-442.

7. White,B.A., Lipczynski,R.T., & Daniels,A.R.
"Real-time control of a linear synchronous motor for use in advanced ground transport"
Real-Time Control of Electromechanical Systems,I.E.R.E. Publication No.58, March 1984, pp49-52.

8. *"M68000 16/32 Bit Microprocessor Programmers Reference Manual."*
Motorola Inc., 1984.

9. Fletcher,W.I.
"An Engineering Approach to Digital Design."
Prentice-Hall Inc., 1980.

10. Richards,M., Aylward,A.R., Bond,P., Evans,R.D., & Knight,B.J.
"TRIPOS—a portable operating system for minicomputers."
Software Practice and Experience (GB), Vol.9,No.7,July 1979,pp513-526.

11. Richards,M., & Whitby-Strevens,C.
"BCPL - the Language and its Compiler."
Cambridge University Press, 1980.

12. Gupta,N.K., Lyons,M.G., Aubrun,J.N., & Margulies,G.
"Modelling, control, and system identification methods for flexible structures."
AGARD 1981, No.260, *"Spacecraft pointing and position control."* pp12-1,12-41

13. Bonvin,D., & Mellichamp,D.A.
"A unified and critical review of modal approaches to model reduction".
International Journal of Control, Vol.35,No.5,1982, pp829-848.

14. Likens,P., Ohkami,Y., & Wong,C.
"Appendage modal co-ordinate truncation criteria in hybrid co-ordinate dynamic analysis."
Journal of Spacecraft, Vol.13,No.10,Oct.1976, pp611-617.

15. Moore,B.C.
"Principal component analysis in linear systems: Controllability, observability, and modal reduction."
I.E.E.E. Transactions on Automatic Control, Vol AC-26,No.1, Feb.1981, pp17-32.